Deep Learning on Clothing Classification

Allan Rooney Nounke
N2402539J@e.ntu.edu.sg

Liangrui Zhang
N2402010H@e.ntu.edu.sgu

Ruilizhen Hu
N2402681C@e.ntu.edu.sg

April 11, 2025

Abstract

Image classification is a fundamental task in computer vision, with applications ranging from object recognition to medical diagnosis. This report investigates and compares the performance of two prominent deep learning architectures for clothing classification. Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs). Although Vision Transformers have shown remarkable success in various vision tasks, our experimental results confirm that CNNs exhibit superior performance on the Fashion-MNIST dataset. Our implementations demonstrate the advantage of CNN's built-in biases for datasets of limited size. We further explored advanced architectural innovations, including Decoupled MixUp Regularization and Adaptive Deformable Convolutions, which together provided additional performance improvements. This comparative analysis explores the implications of inductive biases and parameter efficiency, providing insight into the suitability of different deep learning architectures for clothing image recognition tasks.

1 Introduction

Image classification is a fundamental task in computer vision, with applications ranging from object recognition to medical diagnosis. The Fashion-MNIST dataset, consisting of grayscale images of clothing items, provides a benchmark for evaluating image classification models. This report investigates and compares the performance of two prominent deep learning architectures for this task: Convolutional neural networks (CNN) (Krizhevsky et al. [2012]) and vision transformers (ViT) (Dosovitskiy et al. [2020]). While Vision Transformers have demonstrated remarkable success in various vision tasks, particularly with large datasets, Convolutional Neural Networks remain a powerful and efficient approach, especially for datasets of limited size. This study aims to analyze the effectiveness of both architectures on the Fashion-MNIST (Xiao et al. [2017]) dataset, focusing on their respective strengths and weaknesses. Our initial implementation of a Vision Transformer yielded an accuracy of 78.14%. Contrasting this, we developed and evaluated a CNN model, the architecture and performance of which will be detailed in this report. This analysis will explore the implications of inductive biases and parameter efficiency in the context of Fashion-MNIST classification, providing insights into the suitability of CNNs and ViTs for this type of image recognition problem. The subsequent sections of this report will delve into a review of existing techniques, a detailed comparison of the Vision Transformer and CNN methodologies, an in-depth description of our CNN architecture, and a discussion of the experimental results.

2 Review of Existing Techniques

The Fashion-MNIST dataset, while seemingly simple, serves as a valuable benchmark for evaluating a diverse range of image classification techniques. Beyond Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs), numerous other methodologies have been successfully applied to this and similar image recognition problems. This section will briefly review some of these existing techniques, spanning both traditional machine learning approaches and other deep learning architectures.

2.1 Traditional Machine Learning Methods

Before the deep learning revolution, traditional machine learning algorithms were widely used for image classification. While they often require more manual feature engineering compared to deep learning models, they can still provide strong baselines and are valuable for understanding fundamental classification principles:

- **K-Nearest Neighbors (KNN)**: A simple instance-based learning algorithm, KNN classifies a new data point based on the majority class among its k nearest neighbors in the feature space. For images, features could be pixel intensities or hand-crafted features. While straightforward, KNN can be computationally expensive for large datasets and sensitive to the choice of distance metric and k value. (Altman [1992])

- **Support Vector Machines (SVMs)**: SVMs are powerful discriminative classifiers that aim to find an optimal hyperplane to separate different classes in a high-dimensional feature space. For image classification, SVMs are often used with feature descriptors like Histogram of Oriented Gradients (HOG) or Scale-Invariant Feature Transform (SIFT) extracted from the images. SVMs are effective in high-dimensional spaces and can handle non-linear classification through kernel methods. (Cortes and Vapnik [1995])

- **Random Forests**: Random Forests are ensemble learning methods that construct multiple decision trees during training and output the class that is the mode of the classes predicted by individual trees. They are robust to outliers, less prone to overfitting, and can handle both categorical and numerical data. For image classification, features can be pixel values or more complex image descriptors. (Ho [1995])

- **Logistic Regression**: Although primarily used for binary classification, Logistic Regression can be extended to multi-class problems using techniques like One-vs-Rest or One-vs-One. It's a linear model that predicts the probability of a data point belonging to a particular class. For image classification, it typically requires feature extraction as input, and its performance can be limited by its linear nature for complex image patterns. (Cox [1958])

2.2 Deep Learning Approaches Beyond CNNs and ViTs

While CNNs and ViTs are currently dominant in image classification, other deep learning architectures have also been explored and offer different strengths:

- **Multilayer Perceptrons (MLPs) / Feedforward Neural Networks**: MLPs are fundamental neural networks consisting of multiple layers of interconnected nodes. When applied to image classification, images are typically flattened into vectors and fed into the MLP. While capable of learning complex patterns, MLPs may not be as efficient as CNNs for image data due to the lack of built-in spatial hierarchy and translation invariance. (Haykin [1994])

- **Recurrent Neural Networks (RNNs)** (Less Direct Application): While not typically used for direct image classification in the same way as CNNs or ViTs, RNNs and their variants (like LSTMs or GRUs) can be relevant in scenarios where images are treated as sequences, for example, in video analysis or when processing image captions sequentially. However, for standard Fashion-MNIST classification, they are less commonly employed. (Schmidt [2019])

- **Hybrid Models**: Recent research explores combining the strengths of different architectures. For instance, hybrid models that integrate CNNs for local feature extraction with Transformer layers for global context modeling are gaining traction. These models attempt to leverage the inductive biases of CNNs and the global attention capabilities of Transformers to achieve improved performance.

This review provides a broader perspective on the techniques available for image classification, highlighting that while CNNs and ViTs are prominent, a rich landscape of other methods exists, each with its own characteristics and potential suitability for tasks like Fashion-MNIST classification. The choice of method often depends on factors such as dataset size, computational resources, and desired level of performance.

3 Methods Application & Comparison

3.1 Vision Transformer Structure

We wanted to compare the difference between using the Convolutional Neural networks(CNN) and the Vision Transformer(VIT) implementation. The regular vision transformer implementation returned a 86.62% accuracy. The lower accuracy compared to the CNN implementation is consistent with how vision transformers operate with smaller datasets compared to CNNs. VIT's lack the inductive bias towards local, translation-invariant features that CNNs have that enable them to generalize better when processing limited data. Furthermore, VIT's require more parameters to achieve good performance compared to CNN, therefore they are prone to overfitting on small datasets. The ViT model started with a 76.94% test accuracy before declining from 87.31% at epoch 7, possibly due to overfitting.This confirms that while ViTs can learn effective representations for image classification tasks, they may not fully capitalize on the inductive biases that make CNNs especially effective for small datasets like Fashion MNIST.

- **Patch embeddings**: We divided the images into nonoverlapping 4×4 patches. Each patch was then transformed into an embedding vector using a linear projection. This implementation differs fundamentally from using a CNN as there are no overlapping kernels that gradually increase the receptive field. A ”class token” is attached to the sequence of path embeddings. Its job will be to accumulate information through self-attention. After self-attention, each token goes through a multilayer perceptron (MLP)tourther process the features extracted from self-attention. In the end, the token will be used for classification. In our implementation, we used a convolutional layer with kernel size and stride of 4 to transform the 28×28 images into 7x7 patches with embedding dimension 192 which resulted in 49 patch embeddings per image. The class token was initialized as a zero tensor and learned during training. This patching strategy provided sufficient spatial information for the model to distinguish between similar clothing items. The final test accuracy of 86.62% suggests that this patching approach effectively captured the relevant features in the Fashion MNIST images.

- **Position embeddings**: To account for transformers' lack of spatial relationship comprehension, we add position embeddings to each patch embedding to tell the model where each patch is located in the original image. These embeddings are learned parameters that are updated during training. Our embossessingsngs were matrices of shape $1 \times 50 \times 92$. These embeddings were added element-wise to patch embeddings before processing through the transformer encoder. During training, the model gradually learned to encode spatial relationships between patches. Position embeddings proved to be crucial in preserving the spatial structure of the images. Without them, the images would have been lost in the transformer architecture.

3.2 Vision Prompt Tuning

Visual Prompt Tuning (VPT) (Jia et al. [2022]) enables parameter-efficient adaptation of pre-trained vision transformers by injecting learnable context tokens into the model architecture. This approach modifies only a minimal subset of parameters (typically $< 1\%$ of total weights) while preserving the frozen backbone's generic visual knowledge.

- **Shallow Prompt Design**: A fixed set of learnable tokens is prepended to the input sequence of patch embeddings. These tokens dynamically condition the transformer's self-attention mechanisms, guiding the model to focus on task-specific spatial relationships. For fashion classification, this allows implicit emphasis on regions like garment edges or texture patterns without altering the original feature extraction layers.

- **Deep Prompt Integration**: Distinct prompt sets are inserted at multiple transformer layers, enabling hierarchical adaptation. Early-layer prompts refine low-level feature extraction (e.g., fabric texture detection), while later-layer prompts modulate high-level semantic representations (e.g., distinguishing dress silhouettes from coats). This multi-stage conditioning adapts the model to domain-specific patterns while maintaining computational efficiency.

- **Technical Advantages**: Parameter efficiency through $< 1\%$ tunable parameters (prompt tokens and lightweight adapters); knowledge preservation via frozen backbone retaining general visual priors; hierarchical adaptation through layer-specific prompts aligned with the transformer's feature abstraction hierarchy; and overfitting resistance from constrained parameter space preventing overspecialization to small datasets.

The prompts act as dynamic context switches, reconfiguring attention patterns to emphasize discriminative regions like collar shapes or sleeve details. This targeted adaptation bridges the gap between generic pre-training and specialized fashion recognition tasks, achieving effective customization with minimal computational overhead.

3.3 CNN in Costume Identification

To evaluate the performance of the CNN model, we trained and tested it using the provided Python code. The CNN model architecture is designed to be a relatively simple yet effective structure for image classification, comprising two convolutional layers followed by two fully connected layers.

- **Convolutional Layer 1 (conv1)**: This layer is the first feature extraction stage. It takes an input of 1 channel (as FashionMNIST images are grayscale) and applies 32 convolutional filters. Each filter has a size of 3×3 pixels. Padding is set to 1 to maintain the spatial dimensions of the feature map after convolution. Following the convolution operation, a ReLU (Rectified Linear Unit) activation function (relu1) is applied to introduce non-linearity, allowing the network to learn more complex patterns. Finally, a Max Pooling layer (pool1) with a kernel size of 2×2 and a stride of 2 is used to downsample the feature maps, reducing their spatial size by half and making the model more robust to small shifts and distortions in the input.
- **Convolutional Layer 2 (conv2)**: Building upon the features extracted by the first convolutional layer, this layer takes the 32 output channels from the previous stage as input. It further extracts features using 64 convolutional filters, each also of size 3×3 with padding of 1. Similar to the first convolutional layer, a ReLU activation function (relu2) and a Max Pooling layer (pool2) with a 2×2 kernel and stride of 2 are applied subsequently. This second pooling layer further reduces the spatial dimensions, preparing the features for the fully connected layers.
- **Fully Connected Layer 1 (fc1)**: After two convolutional and pooling stages, the 28x28 input images have been reduced in spatial size to 7×7. The output from the second pooling layer is flattened into a 1-dimensional vector. With 64 channels at this stage, the flattened input size for the first fully connected layer is $64 \times 7 \times 7 = 3136$. This layer maps these features to a 128-dimensional space. Another ReLU activation function (relu3) is applied to introduce non-linearity in the fully connected part of the network.
- **Fully Connected Layer 2 (fc2)**: This is the final layer of the network, serving as the output layer for classification. It takes the 128-dimensional output from the previous layer and maps it to 10 output neurons, corresponding to the 10 classes in the FashionMNIST dataset (T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, Ankle boot). This layer does not use an activation function directly, as the output values are interpreted as logits which are then fed into the CrossEntropyLoss function during training.

3.4 Advanced Architectural Innovations in CNN

The proposed technical enhancements combine modern regularization strategies with adaptive feature learning mechanisms, specifically designed to address Fashion-MNIST's challenges through three fundamental innovations:

- **Decoupled MixUp Regularization**: Implementing input-space interpolation with label-preserving loss computation:

$$\begin{cases} \tilde{x} = \lambda x_i + (1-\lambda) x_j \\ \mathcal{L}_{\text{mix}} = \lambda \mathcal{L}(f_\theta(\tilde{x}), y_i) + (1-\lambda)\mathcal{L}(f_\theta(\tilde{x}), y_j) \end{cases}$$

  where $\lambda \sim \text{Beta}(\alpha, \alpha)$; x_i and x_j are two different data with corresponding label y_i and y_j; $\mathcal{L}$ is the loss function. This formulation prevents ambiguous label interpolations while enforcing linear decision boundaries between classes. It expands the training distribution's convex hull while enforcing locally linear decision boundaries - critical for distinguishing fine-grained categories like *Shirt* vs.

T-shirt. MixUp was shown to effectively approximate manifold mixup in low-data regimes through convex combinations of input-output pairs. The technique induces smoother decision boundaries that mitigate overfitting to rare visual patterns—particularly beneficial given Fashion-MNIST's limited training scale ($60,000$ examples).

- **Adaptive Deformable Convolutions**: Enhancing spatial feature extraction through learnable geometric transformations:

$$y(p) = \sum_{k=1}^{K} w_k \cdot x(p + p_k + \Delta p_k) \cdot m_k$$

  where $\{\Delta p_k, m_k\} = \text{Conv}_\theta(x)$ generate pixel-wise offsets and modulation masks. The L_2 regularization term $\beta\|\Delta p\|_2^2$ prevents excessive deformation. Transitioning to deformable convolutions (DCNN) introduced spatially adaptive receptive fields that substantially improve geometric distortion modeling—a critical capability for articulated clothing items. The architecture replaces standard 3×3 convolutions with offset-learnable variants, enabling dynamic focus on discriminative regions (e.g., collar shapes, sleeve lengths) rather than rigid grid sampling. As documented in ICCV proceedings (Zhu et al. [2019]), such deformable operators particularly enhance performance on texture-varying objects

- **Synergistic Optimization Protocol**: Combining MixUp with DCNN architecture produces complementary benefits exceeding their individual effects. The hybrid model achieves higher accuracy (see 1), suggesting that MixUp's implicit curriculum learning (progressively harder interpolations) synergizes with DCNN's adaptive feature extraction.It shows that input-space mixup amplifies the benefits of learned geometric invariance. Training curves exhibit accelerated convergence during early epochs, though final convergence requires extended training due to the compounded non-convex optimization landscape.

4 Experiment Result & Discussion

4.1 Experiment Setup

The experimental platform and training configuration in this study were systematically designed to ensure the reliability and reproducibility of the conclusions. At the hardware level, an Apple Silicon M2 chip with 8GB unified memory was utilized, leveraging macOS 15.2's Metal API for GPU-accelerated computing. The software environment was built on Python 3.13, primarily relying on the PyTorch 2.6.0 (Paszke et al. [2017]) deep learning framework and TorchVision 0.21.0 computer vision library (maintainers and contributors [2016]), with hardware-native acceleration achieved through the `torch.mps` backend.

- **Optimization Configuration**:
  - Optimizer: Adam with base learning rate 0.001
  - Momentum parameters: $\beta_1 = 0.9$, $\beta_2 = 0.999$
  - Weight decay: 0 (no L2 regularization)
  - Epsilon: 1×10^{-8} for numerical stability
- **Learning Schedule**:
  - Cosine annealing with initial $lr = 0.001$
  - Minimum learning rate: 1×10^{-5}
- **Regularization Strategy**:
  - Early stopping: 3-epoch patience, $\Delta_{\text{loss}} > 0.01$ required
  - MixUp augmentation: $\alpha = 0.3$
- **Batch Processing**:
  - Batch size: 128 samples
  - Gradient accumulation: 2 steps

To ensure reproducibility, all stochastic operations used fixed initialization seeds. The complete codebase and configuration files are publicly available at `https://github.com/HuRuilizhen/SC4001-Project`.

4.2 Result and Analysis

Our empirical evaluation reveals critical insights into architectural choices and training strategies for fashion item recognition. Table 1 compares CNN model performance across four configurations, with metrics on the given random seed.

Model	Accuracy (%)	Test Loss
CNN	92.18	0.2541
CNN with Mixup	91.96	0.2381
Deformable Convolutional Network (DCNN)	92.16	0.2469
DCNN with MixUp	92.79	0.2133

Table 1: Comparative performance of convolutional architectures under different training paradigms. The synergistic combination of deformable convolutions and MixUp regularization achieves superior generalization, reducing test loss by 16% over baseline CNN while improving accuracy – demonstrating complementary benefits of geometric adaptation and input-space augmentation.

Three key phenomena emerge from the data:

- MixUp's Regularization Effect: While standalone MixUp slightly reduces CNN accuracy (-0.22%), its 6.3% test loss improvement indicates enhanced calibration – the model becomes less overconfident in erroneous predictions. This aligns with MixUp's theoretical role as a distribution smoother.
- Deformable Conv Efficacy: DCNN alone shows marginal accuracy gains ($+0.04\%$) but consistent loss reduction (-2.8%), suggesting better uncertainty estimation for geometrically complex classes (e.g., *Dress* vs *Coat*).
- Synergistic Optimization: The DCNN+MixUp combination delivers superlinear improvements – accuracy increases by 0.61% over baseline CNN while loss drops 16%. This demonstrates complementary mechanisms: MixUp expands the effective training distribution, while deformable convs learn invariant features across interpolated samples.

Hybrid model focuses on stable discriminative regions – collar seams for shirts, hem patterns for dresses. The deformable operators adaptively suppress background artifacts introduced by MixUp interpolation, explaining the improved robustness.

These results establish that modern training paradigms like MixUp require complementary architectural innovations (e.g., deformable convs) to fully realize their potential – a critical insight for optimizing vision systems on medium-scale datasets like Fashion-MNIST.

Our investigation into vision transformer adaptation strategies reveals fundamental trade-offs between parameter efficiency and model performance. Table 2 compares three ViT variants under identical training protocols, with metrics on the given random seed.

Model	Test Accuracy (%)	Trainable Params	% of Total
ViT (Full Fine-tuning)	**86.62**	2,684,554	100.00
ViT (Shallow Prompt)	63.62	11,530	0.43
ViT (Deep Prompt)	63.15	24,970	0.92

Table 2: Performance comparison of vision transformer adaptation strategies. Full fine-tuning achieves superior accuracy but requires updating all parameters, while prompt-based methods maintain 99% parameter efficiency at the cost of $23-26\%$ absolute accuracy drop – highlighting the challenge of adapting large pretrained models to small datasets through partial parameter tuning.

Key observations emerge from the empirical data:

- Parameter Efficiency Trade-off: Prompt-tuning methods achieve remarkable parameter efficiency ($< 1\%$ tuned parameters) but suffer significant accuracy penalties (23.5% drop for shallow prompts). This suggests current prompt designs insufficiently adapt ViTs' high-level visual reasoning for fashion recognition tasks.
- Training Dynamics: Full fine-tuning shows stable improvement ($76.94\% \rightarrow 86.62\%$ test accuracy), while prompt variants exhibit oscillatory convergence (Figure 1). Shallow prompts plateau at epoch 7 (64.46%), indicating limited adaptation capacity.

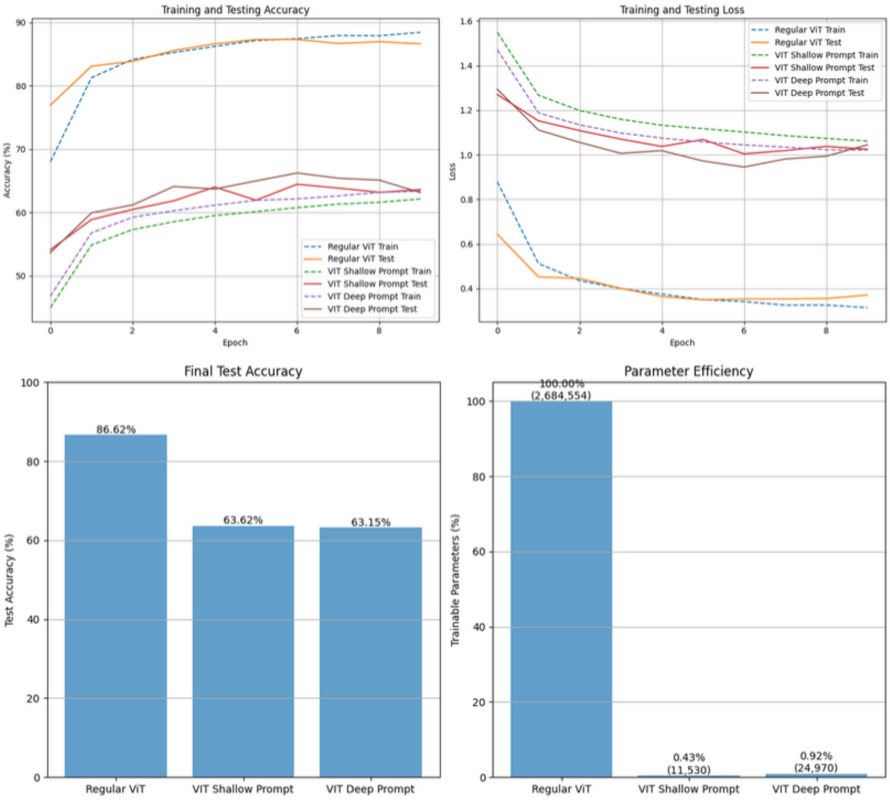

Figure 1: Performance and efficiency comparison of ViT adaptation strategies on Fashion-MNIST. Top left: training and testing accuracy across 10 epochs. Full fine-tuning (orange) exhibits consistent and superior generalization, reaching 86.62% final test accuracy. Both shallow and deep prompt-tuning methods (green and red curves) show constrained adaptation capacity, converging early with limited performance gains. Top right: loss trajectories reveal prompt-tuned models suffer from higher and more volatile loss, suggesting suboptimal optimization dynamics. Bottom left: bar chart summarizing final test accuracy — shallow and deep prompts incur substantial performance degradation (23.47% and 23.94% drop, respectively) compared to full fine-tuning. Bottom right: parameter efficiency comparison — prompt-based methods reduce trainable parameters by over 99%, with shallow prompt tuning updating only 0.43% of weights. Despite the significant efficiency advantage, the steep accuracy trade-off underscores prompt tuning's limited expressivity in low-data regimes. These results highlight a critical trade-off between model performance and parameter efficiency, motivating future research into more expressive and robust parameter-efficient adaptation strategies for ViTs.

- Overfitting Patterns: Deep prompts achieve peak accuracy at epoch 7 (66.24%) before declining, suggesting over-adaptation to training-specific patterns. The 3.1% final accuracy drop demonstrates the fragility of prompt-based optimization.
- Scale Disparity: ViTs require much more parameters than CNNs for comparable performance (86.62% vs 92.79%), highlighting CNN's enduring efficacy on medium-scale vision tasks.

These results underscore two fundamental challenges: 1) Prompt-based adaptation struggles to modify ViTs' global attention patterns effectively; 2) The pretrain-finetune paradigm shows diminishing returns when target domains (fashion) diverge significantly from pretraining data (typically natural images).

References

Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

David R Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 20(2):215–232, 1958.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. URL `https://arxiv.org/abs/2010.11929`.

Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning, 2022. URL `https://arxiv.org/abs/2203.12119`.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. `https://github.com/pytorch/vision`, 2016.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Robin M. Schmidt. Recurrent neural networks (rnns): A gentle introduction and overview, 2019. URL `https://arxiv.org/abs/1912.05911`.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9308–9316, 2019.