Сверточные сети: ResNet (Residual Network)

Диц Днаиил Денисович

3 апреля 2025 г.

Зачем нам ResNet?

Мотивация

- Глубокие сети = лучшее качество (VGG16, VGG19).
- Но с ростом глубины появляются проблемы:
  - **Затухание градиента**
  - **Деградация качества**
- Просто добавлять слои уже недостаточно!

Проблема деградации (Degradation Problem)

- Увеличиваем глубину — а точность **падает**!
- Ошибка обучения становится больше в более глубокой сети.
- Не переобучение, а неспособность обучить! **Сеть не может выучить тождественное преобразование.**

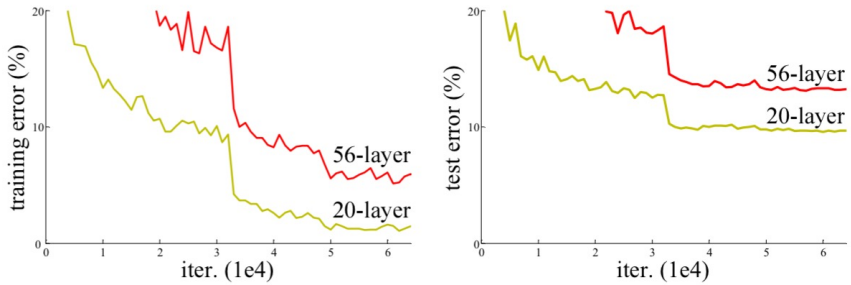

Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

Проблема деградации (Degradation Problem)

- Увеличиваем глубину — а точность **падает**!
- Ошибка обучения становится больше в более глубокой сети.
- Не переобучение, а неспособность обучить! **Сеть не может выучить тождественное преобразование.**

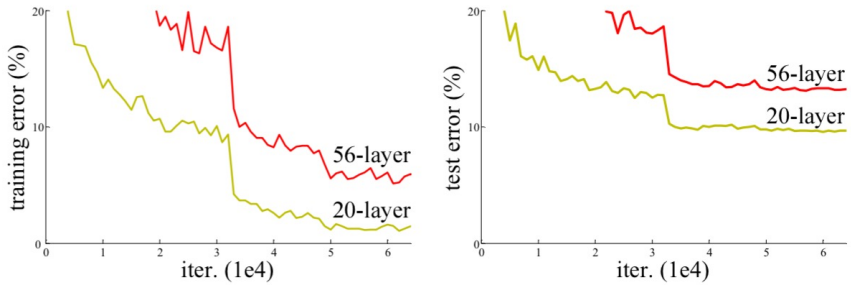

Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

Нужна идея: как передавать информацию напрямую?

ResNet предлагает: Shortcut (skip) соединения

Основная идея ResNet

- Вместо $F(x)$ обучаем **остаточную функцию** $R(x) = F(x) - x$
- Модель учится: $\boxed{y = F(x) + x}$
- Если $F(x) \approx 0$, то $y \approx x$ — передаём информацию напрямую

Основная идея ResNet

- Вместо $F(x)$ обучаем **остаточную функцию** $R(x) = F(x) - x$
- Модель учится: $\boxed{y = F(x) + x}$
- Если $F(x) \approx 0$, то $y \approx x$ — передаём информацию напрямую

Интуиция

"Легче учиться на отклонениях от исходного, чем заново строить всё."

Residual Block — строительный блок ResNet

- Последовательность: Conv $\rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Conv $\rightarrow$ BN
- Затем: $F(x) + x \rightarrow$ ReLU
- **Без увеличения параметров!**

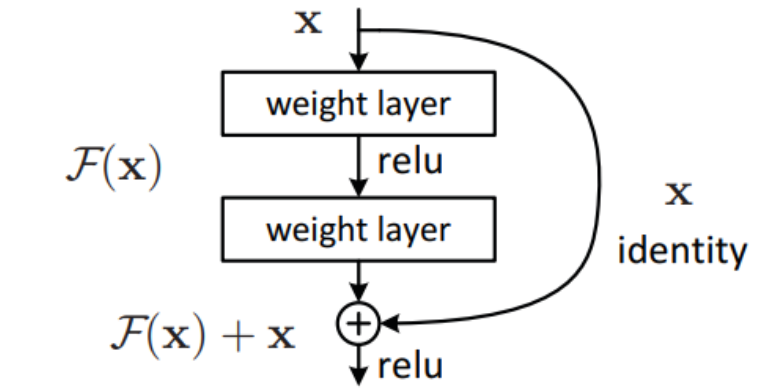

Figure 2. Residual learning: a building block.

Источник: ResNet paper

Почему ResNet помогает с градиентами?

- Backprop для residual блока:

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial F(x)} \cdot \left(\frac{\partial F(x)}{\partial x} + 1 \right)$$

- Появляется **прямой путь** для градиента!

- Градиенты меньше затухают $\Rightarrow$

Почему ResNet позволяет строить очень глубокие сети?

- Shortcut-соединения **не увеличивают параметры**
- Обеспечивают прямой путь для информации и градиентов
- Обучение становится стабильным и быстрым
- Поддержка архитектур с **50, 101, 152 слоев и больше**

Как складывать $F(x) + x$?

- Размерности должны совпадать
- Два способа:
  - **Zero-padding** — добавляем нули в каналы
  - **1x1 свёртка** (projection shortcut) — обучаемо меняем размер

Bottleneck-блоки (для глубоких версий)

- Используются в ResNet-50, 101, 152
- Последовательность:

$$\text{1x1 (уменьшение)} \rightarrow \text{3x3 (обработка)} \rightarrow \text{1x1 (увеличение)}$$

- Эффективно по памяти и скорости
- Сохраняется идея: $F(x) + x$

Пример: использование ResNet из Transformers

Импорт модели и перевод в режим инференса:

```
from transformers import ResNetForImageClassification

model = ResNetForImageClassification.from_pretrained("microsoft/resnet-50")
model.eval()
```

Модель ожидает: изображения размера не менее 224x224,

нормализованные по ImageNet:

- mean = [0.485, 0.456, 0.406]
- std = [0.229, 0.224, 0.225]

Пример: инференс ResNet на изображении

Полный пример классификации с использованием Hugging Face:

```
from transformers import AutoFeatureExtractor, ResNetForImageClassification
import torch
from datasets import load_dataset

dataset = load_dataset("huggingface/cats-image")
image = dataset["test"]["image"][0]

feature_extractor = AutoFeatureExtractor.from_pretrained("microsoft/resnet-50")
model = ResNetForImageClassification.from_pretrained("microsoft/resnet-50")

inputs = feature_extractor(image, return_tensors="pt")

with torch.no_grad():
    logits = model(**inputs).logits

predicted_label = logits.argmax(-1).item()
print(model.config.id2label[predicted_label])
```

Итоги: что дала нам ResNet

Ключевые идеи

- Shortcut-соединения позволяют учить **очень глубокие сети**
- Градиенты не затухают $\Rightarrow$

- Лежит в основе YOLO, ConvNeXt, EfficientNet и других моделей