Resumo: Artigo Backpropagation Applied to Handwritten Zip Code Recognition e comentários postados no blog de Andrej Karpathy

O artigo de 1989 demostra o uso de uma rede neural treinada com backpropagation para uma aplicação de uso real, no caso o reconhecimento de dígitos de códigos postais americanos escritos à mão.

A arquitetura da rede é formada por três camadas escondidas (H1, H2 e H3), sendo as duas primeiras convolucionais, e a última densa. A H1 é formada por 12 feature maps de 8x8 (768 unidades), cada qual recebendo quadrados de 5x5 da imagem como entrada, com pesos compartilhados, além de bias (26 pesos). Totaliza 19.968 conexões (768x26) e 1.068 parâmetros para otimização (768 + 25x12). A H2 é formada por 12 feature maps de 4x4 (192 unidades), cada qual recebendo quadrados de 5x5 dentro de cada um de 8 dos 12 feature maps da camada anterior, mais o bias (201 pesos). Totaliza 38.592 conexões (192x201) e 2.592 parâmetros (192 + 200x12). A H3 possui 30 neurônios totalmente conectados à H2, formando 5790 conexões e parâmetros (30x193). A camada de saída é formada por 10 neurônios totalmente conectados à H3, formando 310 conexões e parâmetros (10x31). Ao total, a rede possui 1.256 neurônios, 64.660 conexões e 9.760 parâmetros independentes.

Como função de ativação usou-se tangente hiperbólica, e como função de perda o erro quadrático médio (MSE). Os pesos foram inicializados randomicamente sob uma distribuição uniforme. Para otimização dos pesos, utilizou o gradiente estocástico, após a predição de cada exemplo. Após 23 épocas, foi obtido um erro de 0,14% das classificações no conjunto de treinamento e 5,0% no conjunto de teste.

O modelo proposto representou um avanço, por permitir um uso de rede neural formada por camadas convolucionais contendo pesos aprendidos ao invés aplicar engenharia de features, e uso de backpropagation. Conforme comentou Karpathy no blog, as especificações discutidas no paper são totalmente relevantes ainda hoje: o dataset, a arquitetura da rede, a função de perda, a otimização, e as taxas de erros das classificações nos conjuntos de treinamento e teste. A implementação sim, mudou muito, hoje sendo organizada em três componentes: uma biblioteca de operações matemáticas de tensores; um motor autograd, que registra o grafo computacional no foward pass da rede e automatiza o backward pass; e uma API de alto nível para montar as redes neurais, operações, otimizações, funções de perda etc.

Sobre a mesma rede, Karpathy implementou alguns ajustes: alteração da saída da rede para softmax e função de perda entropia cruzada; mudança do mecanismo de otimização para AdamW; aumento de dados deslocando imagens originais em 1 pixel horizontal ou verticalmente; dropout de 25% dos neurônios antes da camada H3; alteração das funções não lineares das camadas escondidas para ReLU; e aumento para 80 épocas. Obteve erro de 1,47% em treinamento e 1,59% em teste. Além disso, trocando a base de dados original pela base MNIST que temos hoje, o conjunto de treino aumentou de 7.291 imagens para 50 mil. Chegou-se assim a erro de 1,07% em treinamento e 1,25% em teste. O tempo de treinamento em 1989 foi de 3 dias em uma workstation Sun, enquanto o atual em um laptop com GPU demorou apenas 90 segundos. A alteração da estratégia de otimização de estocástica para full-batch permitiu o uso máximo da GPU, acelerando o treinamento em mais 100 vezes.

Finalmente, o post comenta como maiores alterações estruturais em relação ao que LeCun e colegas fizeram em 1989 e o que se fazia até pouco tempo o ajuste fino em modelos pré-treinados e o uso dos modelos neurais fundacionais como Bert, T5 e GPT-3.