Advanced Institute for Artificial Intelligence

PyTorch e Keras

Advanced Institute for Artificial Intelligence – AI2

https://advancedinstitute.ai

PyTorch

Agenda

- Frameworks para Desenvolver Redes Neurais
- Elementos básicos do PyTorch
- Exemplos de Comandos e Redes Neurais em Pytorch
- Keras Sequential

PyTorch

Pytorch e Keras (módulo interno ao Tenorflows)

- Frameworks para desenvolvimento de redes neurais
- Apresentam maior abstração para uso mais fácil de estrutura de dados do tipo tensores
- Flexibilidade para implementar também o baixo nível do treino da rede neural
- São frameworks equivalentes

PyTorch

Elementos básicos para criar uma rede neural em PyTorch

- Tensores: estrutura fundamento de dados do PyTorch
- Dataloader: classe para controlar a recuperação de dados, durante o treinamento do modelo
- Dataset: classe para controlar a manipulação de dados
- Autograd engine para cálculo diferencial
- Module: Classe base para todos os módulos de rede neural

PyTorch

O Tensor do PyTorch:

- Similar a estrutura do Tensorflow
- Interoperabilidade com numpy
- Permite coordenar o uso de CPUs e GPUs, direcionando os tensores entre os dispositivos

Named Tensors

- Funcionalidade do PyTorch que permite associar a cada coluna do Tensor uma identificação
- Tais identificações podem ser usadas para realizar operações sem que seja necessário conhecer a dimensão que a informação se encontra
- Essa funcionalidade permite associar uma grau de abstração ao código que aproxima dos conceitos conhecidos do domínio

PyTorch

dataset e dataloader

- Dataset: permite implementar rotinas específicas de acesso aos dados, de acordo com a natureza do problema
- DataLoader: recurso de alto nível do PyTorch para realizar leitura de dados e submeter ao processo de treinamento
- Para uma bases grandes, o dataset e o dataloader permitem acessar os dados que serao utilizados na época, sem comprometer a capacidade de memória e processamento

Keras

Framework Keras

- Objetivo Sequential é o elemento base para criar uma rede neural
- O sequential permite desenhar a arquitetura da rede neural por meio do comando add
- O comando compile define a função loss, otimizador e a métrica
- O comando fit treina o modelo com base nos parâmetros
- O objeto history é gerado pelo fit e contém a informação de loss e da métrica de desempenho para cada época

Keras

- Keras pode ser montado com suporte de Tensorflow, Theano e CNTK
- Keras oferece suporte quanto ao uso de várias GPUs e treinamento distribuído (Horovod)
- Os modelos Keras podem ser transformados em estimadores de TensorFlow e treinados em clusters de GPUs no Google Cloud
- Keras pode ser executado no Spark via Dist-Keras (do CERN) e Elephas

Keras

- O modelo precisa saber que formato de entrada será recebido. Esse formato é definido na primeira camada em um modelo Sequential, as camadas subsequentes definem a entrada e saída de acordo com os parâmetros da Rede Neural
- Como definir o formato de entrada:
  - Argumento inputshape na primeira camada. Essa é uma tupla de forma (uma tupla de números inteiros ou Nenhuma, em que Nenhuma indica que qualquer número inteiro positivo pode ser esperado)
  - Algumas camadas 2D, como Dense, permitem a especificação de sua forma de entrada por meio do argumento inputdim, e algumas camadas temporais 3D suportam os argumentos inputdim e inputlength

Keras

Configurando o método de compilação.

- optimizer (Otimizador). Pode ser o identificador de sequência de um otimizador existente (como por exemplos rmsprop ou adagrad) ou uma instância da classe Optimizer
- loss (função de perda). Esse é o objetivo que o modelo tentará minimizar. Pode ser o identificador de cadeia de uma função de perda existente (como categoricalcrossentropy ou mse) ou pode ser uma função objetiva
- métrics (lista de métricas). Para qualquer problema de classificação, você desejará definir isso como métricas = ['precisão']. Uma métrica pode ser o identificador de sequência de uma métrica existente ou uma função de métrica personalizada

Keras

Treinando um modelo

- Os modelos Keras são treinados em matrizes Numpy de dados e rótulos de entrada
- Para treinar um modelo, você normalmente usa a função fit
- A função fit gera um objeto history que pode ser usado para realizar o plot da curva de loss