03

Para saber mais

Nesta aula, tivemos uma visão geral de manipulação de tensores, mas ainda há um universo de possibilidades que pode ser encontrado na documentação do objeto Tensor do PyTorch (https://pytorch.org/docs/stable/tensors.html).

Uma operação muito útil, que não faz parte diretamente do módulo de tensores, é a concatenação. Modelos mais complexos, que envolvem fusão de informação, ou entradas que precisam ser manipuladas antes de ser enviadas para a rede, se beneficiam muito dessa operação. Para isso, o pacote `torch` traz a função cat (https://pytorch.org/docs/stable/torch.html#torch.cat) que tem o seguinte padrão:

```
tns_out = torch.cat( (tns1, tns2), dim=0 )
```

Ou seja, a função recebe um objeto tipo tupla contendo o conjunto de tensores a ser concatenados, seguido da dimensão de concatenação. Os tensores devem ter dimensões idênticas, exceto na dimensão de concatenação.

Outras operações para combinar múltiplos tensores pode ser encontrada nessa parte da documentação (https://pytorch.org/docs/stable/torch.html#indexing-slicing-joining-mutating-ops).

Para trabalhar com Deep Learning no PyTorch é preciso dominar a arte de manipular tensores e transformá-los da forma que o problema necessitar. Explore amplamente a documentação do pacote torch (https://pytorch.org/docs/stable/torch.html) e encontrará funções como `torch.squeeze()` e `torch.unsqueeze()` que, respectivamente, removem e adicionam dimensões de tamanho 1. Essas e outras funções vão facilitar muito o seu trabalho e te tornar um mestre na arte dos tensores!