View vs Reshape in Pytroch

For the Russian version scroll down below

`torch.view` and `torch.reshape` are responsible for essentially the same role: morphing the shape of the tensors into desirable dimensions.

The main difference is that `torch.view` imposes some **contiguity** constraints, while `torch.reshape` doesn't.

It leads to some dissimilarities in function behavior:

1. `view()` returns a new view to the same tensor, e.g. it maps to the same memory address, and any changes done to such tensor affect the original one and vice versa.
2. `reshape()` sometimes returns the same tensor, and sometimes clones elements of the original tensor to a new memory heap.

Thus, `view()` is often a faster, memory-efficient way of changing the tensor shape, although it may cause errors due to contiguity constraints, and may need `.contiguous()` to resolve this issue. On the other hand, `reshape()` is slower and more stable.

https://stackoverflow.com/questions/49643225/whats-the-difference-between-reshape-and-view-in-pytorch

view или reshape?

В торче функции `torch.view` и `torch.reshape` ответственны примерно за одно и то же: изменение размерности тензора, не меняя при этом сам тензор.

Однако `view()` накладывает ограничение на смежность данных (**continguity**), тогда как `reshape()` этого не делает.

Это приводит к следующей разницы в поведении:

1. `view()` возвращает тот же тензор с измененной размерностью, обращаясь к тому же адресу в памяти
2. `reshape()` иногда может вернуть новый тензор, а иногда - изменить размерность исходного.

В итоге, `view()` работает быстрее, но может спотыкаться из-за несмежности данных. Поэтому обычно рекомендуют использовать `reshape()`

https://stackoverflow.com/questions/49643225/whats-the-difference-between-reshape-and-view-in-pytorch