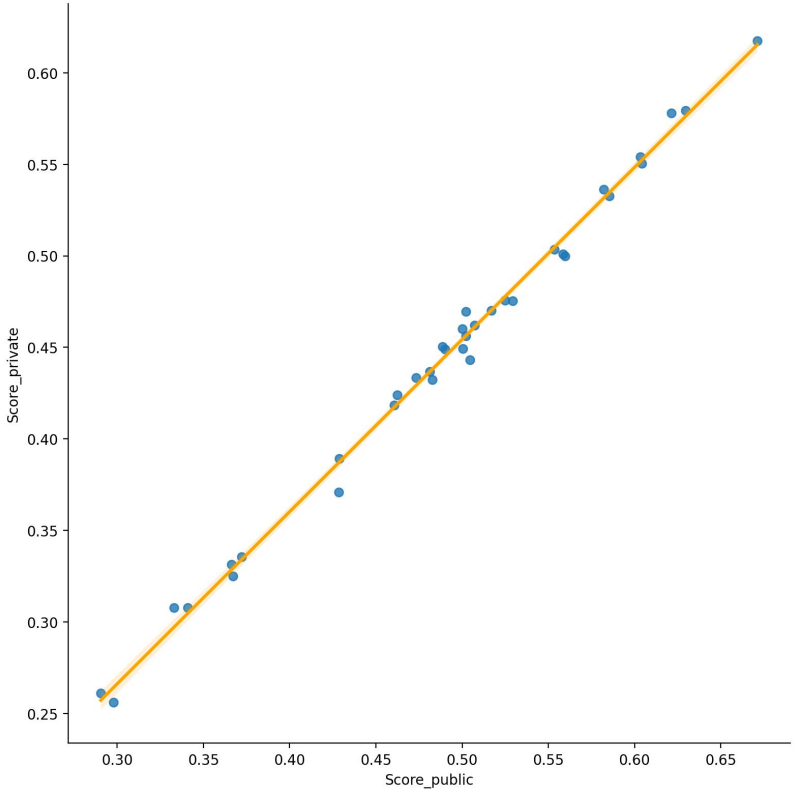
Как решать задачи

Без боли и страданий

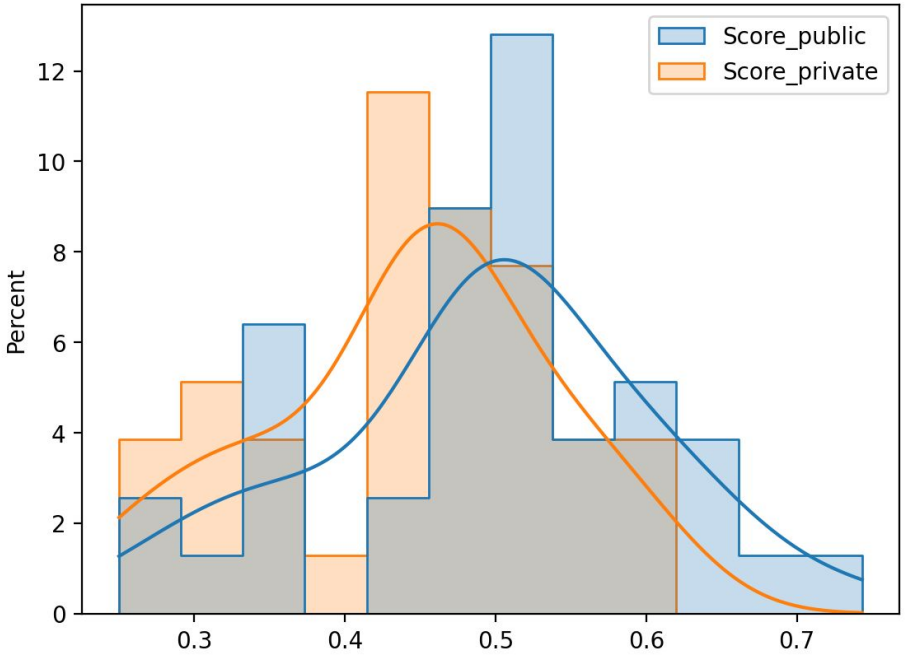
План

1. Результаты соревнования
2. Доклады (5 минут презентация + 3-5 вопросы)
3. Best practices

Лидерборд HW2



	Median	Max
Public	50.05	74.33
Private	44.92	61.75



Пайплайн решения ML-задачи

1. Постановка задачи
2. Данные
3. Модель
4. Обучение
5. Оптимизация
6. Деплой и поддержка

- Чаще всего в виде абстрактной проблемы
- Перевод с продуктового языка на метрический: как померить качество получившегося решения

Пайплайн решения ML-задачи

1. Постановка задачи
2. Данные
3. Модель
4. Обучение
5. Оптимизация
6. Деплой и поддержка

- ТЗ на разметку
- Шумная разметка / domain adaptation
- Синтетика
- Препроцессинг максимально похож на претрейн
- train и val аугментации должны быть похожи (кроме TTA)
- Аугментации адекватны таргету

Пайплайн решения ML-задачи

1. Постановка задачи
2. Данные
3. Модель
4. Обучение
5. Оптимизация
6. Деплой и поддержка

- Бесполезный Ad-Hoc
- Baseline: сильный переобучится под шумную разметку, слабый не перформит
- Softmax <-> NLLLoss, сложные активации и лоссы
- Инициализация нулями (CE: ~ log K)

Пайплайн решения ML-задачи

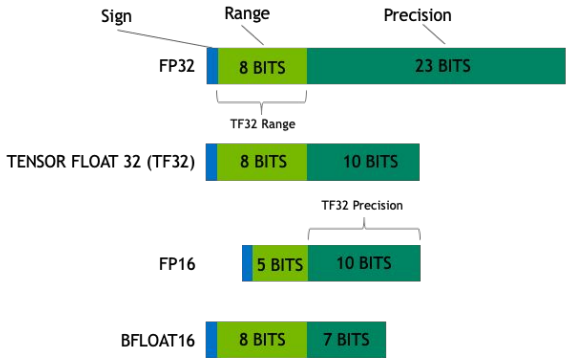
1. Постановка задачи
2. Данные
3. Модель
4. Обучение
5. Оптимизация
6. Деплой и поддержка

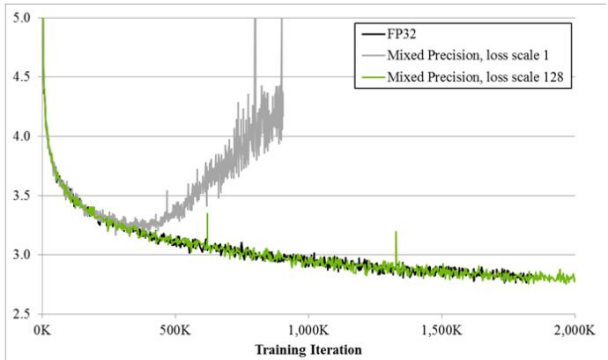
- One Batch Overfitting
- zero_grad
- model.train()/model.eval() (BatchNorm)
- loss.item() (или detach) между итерациями
- torch.save(optim.state_dict())
- 1 изменение за раз
- Не учить короткие экспы до сходимости
- Адекватность прироста качества (не шум)
- Лосс как прокси метрики
- LRFinder: геометрическая прогрессия до взрыва
- Всё нужно логгировать и отсматривать

Пайплайн решения ML-задачи

1. Постановка задачи
2. Данные
3. Модель
4. Обучение
5. Оптимизация
6. Деплой и поддержка

- torch.compile()/TensorRT/ONNX
- Прунинг/Квантизация/Дистилляция (позже)
- Mixed precision training



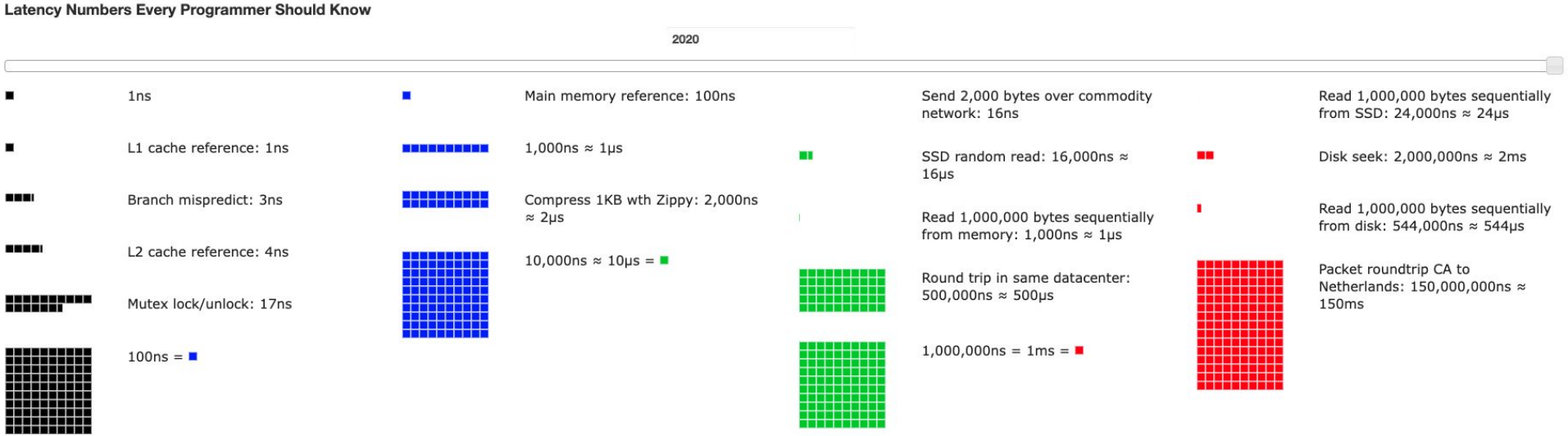


Пайплайн решения ML-задачи

1. Постановка задачи
2. Данные
3. Модель
4. Обучение
5. Оптимизация
6. Деплой и поддержка

- Как в SWE: unit-тесты, мониторинг, A/B
- Data Drift, часто детектится отдельной моделью
- Унифицированный препроцессинг
- Эффективный (асинхронный) инференс
- Постпроцессинг и чистка предиктов, Active Learning
- Профилирование кода

Ботлнеки в загрузке данных

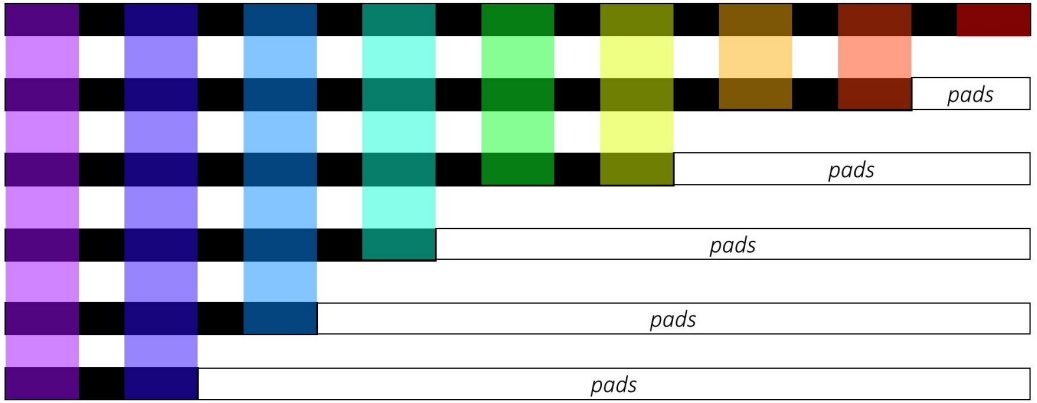


https://colin-scott.github.io/personal_website/research/interactive_latency.html

Препроцессинг

- Что и как читать
- num_workers > 1 (кэп)
- Pillow-SIMD: drop-in замена PIL
- Faster JPEG decoders
- CPU bound: перенос тяжёлых аугов на GPU (DALI) или в офлайн
- Последовательности: прегенерация и умный паддинг

Padded sequences sorted by decreasing lengths



Профилирование

- White-box оптимизация пайплайна обучения и инференса
- DeepSpeed FLOPs breakdown
- PyTorch Profiler