Отчёт по выполнению 2 домашнего задания

Сендерович Александра Леонидовна, ФКН

1 Сплит и обучение обычной модели

Для начала необходимо было зафиксировать разделение данных на обучение и валидацию, чтобы можно было сравнивать результаты разных экспериментов. Для этого я сгенерировала и сохранила в файл случайную перестановку индексов данных. Далее первые 80 процентов индексов в этом порядке относятся к обучающей выборке, остальное – к валидации. После этого на данных из обучающей выборки я обучила бейзлайн и получила качество примерно 2.9e-05 на валидации. Эта модель далее была использована как учитель при дистилляции.

2 Стриминг

Для реализации стриминга пришлось написать новую архитектуру сети, не такую, как в бейзлайне. Разница в том, что Attention в стриминге должен использовать предыдущие энергии, не пересчитывая их заново, а также возвращать новую посчитанную энергию для нового обработанного фрейма. Кроме новой архитектуры, я написала дополнительный класс, наследующийся от nn.Module, для применения стриминга. В нём как раз и была заложена основная логика: мы добавляем в рассмотрение по одному таймстемпу; если с учётом страйда можно применить свёртку, применяем её и далее GRU, используя hidden_state с прошлого шага. К новому результату GRU и $T - 1$ старым (T – число свёрток со страйдом, помещающихся в размер окна) применяем Attention, используя при этом $T - 1$ сохранённых энергий. Если же свёртка с учётом страйда не помещается, просто дублируем предыдущее полученное значение. Таким образом, на каждом шаге необходимо хранить информацию всего о T предыдущих элементах.

В ноутбуке приведены примеры работы стриминга на двух аудиозаписях. Первая состоит из аудиозаписей, встречающихся в валидации: 5 записей без ключевого слова + 1 с + 5 без. Вторая состоит из шума из стороннего датасета и ключевого слова из валидации. На обоих модель хорошо распознаёт ключевое слово, но на обоих при этом есть и другие, более низкие пики. Также краткий комментарий о результатах экспериментов дан и в самом ноутбуке.

3 Сжатие и ускорение

Здесь я попробовала три сетапа. В первом я использовала только Dark Knowledge Distillation. Здесь я потратила много времени на перебор гиперпараметров, поставив около 25 экспериментов, чтобы качество ученика стало лучше качества бейзлайна. При этом сначала я сравнивала качество ученика с качеством учителя, а не бейзлайна, написанного в задании, что делало задачу сложнее. В итоге у меня получилась модель из 9047 параметров, которую пришлось тренировать 50 эпох и которая давала качество 3.7e-5 на валидации. С помощью модуля thop удалось выяснить, что такая модель быстрее модели учителя примерно в 5.3 раза и занимает в 6.5 раз меньше места.

Далее я реализовала Attention Distillation, где вместе со стандартным лоссом используется KL-дивергенция между выходами слоёв внимания модели учителя и ученика. Чтобы это давало улучшение по памяти и времени, я также реализовала новый слой Attention, в котором внутренняя размерность задаётся как отдельный параметр, а не равна просто размеру скрытого состояния из модели. Я применила Dark Knowledge и Attention Distillation и получила модель с 8937 параметрами, для обучения которой уже потреовалось 70 эпох. Профилирование показало, что такая модель быстрее снова примерно в 5.3 раза, а меньше – в 6.6. При этом качество ухудшилось по сравнению с прошлым сетапом: новое качество составило примерно 5.2e-0.5 (но это всё ещё меньше 5.5e-5!).

В третьем сетапе я применила к полученной во втором сетапе модели dynamic quantization. Здесь оказалось неожиданным, что квантизация работает только для моделей на CPU (по крайней мере, в колабе), а также то, что thop неправильно считает время для таких моделей. Поэтому время применения получившейся модели я считать не стала, решив, что квантизация влияет лишь на размер. А по размеру полученная модель стала примерно в 16 раз меньше. Качество ухудшилось в третьем знаке, но с точностью до второго осталось равным примерно 5.2e-05.

В итоге мои улучшения: 5.3 раза по времени и 16 раз по памяти. К сожалению, из-за нехватки времени продолжить работу над ускорением модели не удалось. Однако основное улучшение приносит дистилляция: скорее всего, для получения ускорения в 10 раз нужно довольно долго подбирать гиперпараметры (температуру и вес лосса дистилляции) для обучения очень маленькой модели.