Дистилляция моделей и данных

Баринов Никита
МФТИ

Филатов Андрей
МФТИ

Abstract

Замечено, что во многих задачах ML точность предсказания модели зависит от её размера. При этом зачастую данная зависимость выглядит достаточно тривиально: последовательное увеличение размеров модели позволяет последовательно улучшать точность её предсказаний. Но такой безграничных рост приводит к ряду проблем: существенное увеличение времени обучения, повышенные аппетиты таких моделей к размерам и качеству обучающей выборки, а также вычислительные сложности. Аналогичная проблема и с данными: чем их больше, тем дольше модель на них обучается. В связи с этим возникает желание одновременно "сжать"и данные, и модели так, чтобы на новых данных модель меньшего размера не сильно теряла в качестве. Этот процесс называется дистилляцией моделей и данных, и в статье мы предлагаем одно из решений. Проблемой является то, что при любом "сжатии"данных или моделей теряется информация, поэтому сохранение качества невозможно, но мы можем максимально уменьшить эти потери. Вычислительные эксперименты проводятся на выборках изображений рукописных цифр MNIST.

Keywords Deep Learning · Distilling the Knowledge · Dataset Distillation · Model Compression

1 Introduction

Глубокое обучение добилось огромного успеха за последние несколько лет в различных областях, таких как компьютерное зрение, обработка естественного языка и распознавание речи. Но что же стоит за этими на первый взгляд не очень сложными словами? Обработка огромных объемов информации, тысячи часов работы GPU, сложные и тяжёлые модели - всё это скрывается за словосочетанием "глубокое обучение".

Со временем начали появляться методы "сжатия"моделей без сильной потери качества. Тут появился термин дистилляция знаний(knowledge distillation) – это способ обучения в первую очередь нейросетевых моделей машинного обучения, направленный на передачу знаний от модели-учителя к модели-ученику. Данная проблема происходит из следующих соображений. Много раз было замечено, что в широком диапазоне практически значимых задач машинного обучения точность предсказания модели существенно зависит от её размера. При этом зачастую данная зависимость выглядит очень просто: последовательное увеличение размеров модели позволяет последовательно улучшать точность её предсказаний. Однако такой безграничный рост приводит к ряду проблем, связанных с практическим применением итоговых моделей. Сюда относятся рост времени обучения больших моделей и строгие запросы таких моделей к размерам и качеству обучающей выборки. Кроме того, большие модели нередко требуют более дорогостоящего вычислительного оборудования для эффективного применения, особенно если мы говорим об обработке большого количества запросов в сжатые сроки. А для некоторых сценариев, таких как предсказание в реальном времени и/или на мобильных устройствах, применение большой модели может оказаться вовсе невозможным. Первой статьёй, в которой можно встретить дистилляцию знаний в современном виде является [1].

Несколькими годами позже появился термин дистилляция данных(dataset distillation) - это существенное уменьшение выборки, путём создания искусственных объектов (синтетических данных), которые агрегируют полезную информацию, хранящуюся в данных, и позволяют настраивать алгоритмы

машинного обучения не менее эффективно, чем на всех данных. Но почему дистилляция наборов данных полезна, ведь кажется, что мы теряем большую часть информации? Существует чисто научный вопрос о том, сколько данных закодировано в данном обучающем множестве и насколько оно сжимаемо? Итак, если мы имеем лишь несколько достаточно хорошо дистиллированных изображений, мы можем гораздо эффективнее обучить нейронную сеть на целом наборе данных, по сравнению с традиционным обучением, при котором часто используются десятки тысяч шагов градиентного спуска. Каждый элемент синтетических данных содержит в себе больше информации, чем отдельный элемент исходного датасета. Например, в нашей работе на выборке MNIST каждому классу соответствует ровно один элемент дистиллированного датасета.

В нашей статье предлагается новый подход: одновременная дистилляция модели и данных методами [1] и [2]. Мы попробуем сначала сжать информацию, затем обучить на новых синтетических данных большую модель, а затем дистиллировать её. Мы подробно поставим оптимизационную задачу, посмотрим применение предложенного метода в реальных задачах, сравним подходы. Для анализа качества предложенного решения приводится вычислительный эксперимент на выборке MNIST. (потом пару предложений о полученных результатах)

2 Related works

Сегодня существует несколько решений проблемы дистилляции моделей или данных в отдельности.

2.1 Дистилляция моделей

В [3] говорится, что активации, нейроны или особенности слоев могут также использоваться для обучения меньшей модели. Есть ещё один способ дистилляции, описанный в статье [4], он основан на том, что меньшая модель(ученик) имитирует большую(учитель), тем самым получается конкурентоспособная производительность. В [1] применили дистилляцию, чтобы сжать ансамбль в одну модель. Одной из последних работ является [5]. В ней описывается дистилляция в онлайн-режиме: модель и ученик совместно оптимизируются на каждой итерации. Также существует кросс-модельная дистилляция(передача знаний между промежуточными моделями), одним из сценариев котороя является [6]: имеется граф взаимоотношений между моделями, а передача знаний осуществляется при помощи предложенной функции потерь, сохраняющей локальность.

2.2 Дистилляция данных

Самым простым вариантом может быть оптимизация численными методами. Например, сначала данные инициализируются случайным шумом, а затем при помощи градиентного спуска происходит обновление синтетических данных. Эта процедура подробнее описана в [7]. Описанный метод имеет явный недостаток: он ограничен числом эпох обучения. Использование теоремы о неявной функции в [8] помогает избавиться от такого недостатка. В [9] в качестве функции ошибки используется расстояние между градиентами этой ошибки по параметрам ученика, которые получаются при обучении на обычных и дистиллированных данных. Альтернативным вариантом может быть введение генеративной модели, способной из шума и меток класса создавать необходимые для обучения синтетические изображения, этот подход подробно описан в [10]. Статья [2] предлагает метод дистилляции путем создания датасета, на котором динамика обучения такая же, как и на исходном датасете.

3 Постановка проблемы

Пусть $\mathcal{D}_{real} = \{x_i\}_{i=1}^{N}$ - исходный датасет. Наша задача - создать меньший датасет $\mathcal{D}_{syn} = \{x_i\}_{i=1}^{M}$, где $M \ll N$ и такой, что качество модели, обученной на нём похоже на качество при обучении на исходных данных. Наш метод дистилляции предполагает создание экспертных траекторий обучения τ^*, под которыми понимается последовательность параметров $\{\theta_t^*\}_{t=0}^{T}$, полученных во время убучения нейронной сети на $\mathcal{D}_{real}$. Чтобы получить экспертные траектории, мы обучим большое количество нейронных сетей на $\mathcal{D}_{real}$ и сохраним их параметры на каждой эпохе. Также определим $\hat{\theta}_t$ - параметры модели-студента, обученной на $\mathcal{D}_{syn}$ на шаге обучения t. На каждом шаге обучения мы будем выбирать случайно θ_t^*, инициализировать этим значением парметры модели-студента $\theta_t^* := \theta_t^*$. Установим верхнюю границу T^{max} на число t, чтобы игнорировать ту часть обучения, где параметры меняются незначительно.

Пусть $l(\mathcal{A}(\mathcal{D}_{syn}), \theta_t)$ - дифференцируемая функция потерь, $\mathcal{A}$ - дифференцируемая техника аугментации данных [3]. После инициализации параметров модели-студента мы совершим N шагов градиентного спуска по параметрам $\hat{\theta}_t$:

$$\hat{\theta}_{t+n+1} = \hat{\theta}_{t+n} - \alpha \nabla l(\mathcal{A}(\mathcal{D}_{syn}), \hat{\theta}_{t+n}), \tag{1}$$

где α - шаг обучения модели-студента, используемый для обновления её параметров. После обучения градиентного спуска для конкретной траектории $\tau^* \in \{\tau_i^*\}$ считаем

$$\mathcal{L} = \frac{\| \hat{\theta}_{t+N} - \theta^*_{t+M} \|_2^2}{\| \theta^*_t - \theta^*_{t+M} \|_2^2}, \tag{2}$$

где $\mathcal{L}$ - функция потерь между конечными параметрами студента и учителя, нормированная на пройденное учителем расстояние, что помогает получать информацию о более поздних стадиях его обучения, где параметры меняются не сильно. В конце мы обновялем $\mathcal{D}_{syn}$ в соответствии с обучаемым параметром α и посчитанной функцией $\mathcal{L}$. Итоговый алгоритм выглядит так:

Algorithm 1: Data Distillation

Data: $\{\tau_i^*\}$ – множество параметров учителей, обученных на $\mathcal{D}_{real}$
Data: M - число обновлений между стартовыми и целевыми параметрами учителя
Data: N - число обновлений студента за один шаг дистилляции
Data: $\mathcal{A}$ – дифференцируемая функция аугментации
Data: $T^{max} < T$ – максимальная стартовая эпоха
Result: Дистиллированный набор $\mathcal{D}_{syn}$ и α
$\mathcal{D}_{syn} \leftarrow \mathcal{D}_{real}$;
$\alpha \leftarrow \alpha_0$;
for $step : 1 .. N$ do
  $\tau^* \sim \{\tau_i^*\}, \tau^* = \{\theta_t^*\}_0^T$ - выбираем траекторию обучения;
  $t \le T^*$ - случайно выбираем начальную эпоху;
  $\theta_t^* := \theta_t^*$ - инициализируем веса студента параметрами учителя;
  for $n : 0 .. N-1$ do
    $b_{t+n} \sim \mathcal{D}_{syn}$ - выбрать мини-батч из $\mathcal{D}_{syn}$;
    $\hat{\theta}_{t+n+1} \leftarrow \hat{\theta}_{t+n} - \alpha \nabla l(\mathcal{A}(\mathcal{D}_{syn}), \hat{\theta}_{t+n})$;
  end
  $\mathcal{L} \leftarrow \| \hat{\theta}_{t+N} - \theta^*_{t+M} \|_2^2 \ / \ \| \theta^*_t - \theta^*_{t+M} \|_2^2$;
  Изменить $\mathcal{D}_{syn}$ и α в зависимости от $\mathcal{L}$;
end

Итого первая оптимизационная задача выглядит так:

$$\hat{\mathbf{x}}, \alpha = \arg\min_{\mathbf{x}, \alpha} l(\mathbf{x}, \alpha, \theta), \ \mathcal{D}_{syn} = \bigcup_{\hat{x} \in \hat{\mathbf{x}}} \hat{x}. \tag{3}$$

Далее стоит задача обучения нейросети на дистиллированных данных $\mathcal{D}_{syn}$ и дистилляция модели.

Def 1 : Дистилляция модели - снижение сложности модели путем выбора модели в множестве более простых моделей на основе анализа пространства параметров и предсказаний целевой перменной более сложной фиксированной модели.

Def 2 : Учитель - фиксированная модель, ответы которй используются при выборе модели-ученика.

Def 3 : Ученик - модель, которая выбираемся согласно заданному критерию качества учителя.

Итак, решается задача класификации:

$$\mathcal{D} = \{(\hat{x}_i, y_i)\}_{i=1}^R,$$

где $y_i \in \mathbb{Y} = 1, 2, ..., R$, $\hat{x}_i \in \mathbb{R}^n$.

В дистилляции Хинтона [1] рассматривается параметрическое семейство функций:

$$\mathcal{G} = \{\mathbf{g} \mid \mathbf{g} = softmax(\mathbf{z}(\mathbf{x})/T),\ \mathbf{z} : \mathbb{R}^n \to \mathbb{R}^R\}, \tag{4}$$

где $\mathbf{z}$ - дифференцируемая параметрическая функция заданной структуры, T - параметр температуры. В качестве модели-учителя рассматривается функция $\mathbf{f}$ из множества:

$$\mathcal{F} = \{\mathbf{f} \mid \mathbf{f} = softmax(\mathbf{v}(\mathbf{x})/T,\ \mathbf{v} : \mathbb{R}^n \to \mathbb{R}^R\}, \tag{5}$$

где $\mathbf{z}$ - дифференцируемая параметрическая функция заданной структуры, T - параметр температуры.

При этом температура T имеет свойства:

1. при T $\to 0$ получаем вектор, в котором один из классов имеет единичную вероятность;
2. при T $\to \infty$ получаем вектор, в котором все классы равновероятны.

Функция потерь $\mathcal{L}$ учитывает перенос инфорации от модели-учителя $\mathbf{f}$ к ученику $\mathbf{g}$ и имеет вид:

$$\mathcal{L}(\mathbf{g}) = -\sum_{i=1}^{m}\sum_{r=1}^{R} y_i^r \log \mathbf{g}(\mathbf{x_i})\Big|_{T=1} - \sum_{i=1}^{m}\sum_{r=1}^{R} \mathbf{f}(\mathbf{x_i})\Big|_{T=T_0} \log \mathbf{g}(\mathbf{x_i})\Big|_{T=T_0}, \tag{6}$$

где первое слаагемое отвечает за исходную функцию потерь, а второе - за дистилляцию. Итого получаем оптимизационную задачу:

$$\hat{\mathbf{g}} = \arg\min_{\mathbf{g}\in\mathcal{G}} \mathcal{L}(\mathbf{g}). \tag{7}$$

Список литературы

[1] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015.

[2] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4750–4759, 2022.

[3] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. arXiv preprint arXiv:1412.6550, 2014.

[4] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? Advances in neural information processing systems, 27, 2014.

[5] Inseop Chung, SeongUk Park, Jangho Kim, and Nojun Kwak. Feature-map-level online adversarial knowledge distillation. In International Conference on Machine Learning, pages 2006–2015. PMLR, 2020.

[6] Hanting Chen, Yunhe Wang, Chang Xu, Chao Xu, and Dacheng Tao. Learning student networks via feature embedding. IEEE Transactions on Neural Networks and Learning Systems, 32(1):25–35, 2020.

[7] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. arXiv preprint arXiv:1811.10959, 2018.

[8] Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In International Conference on Artificial Intelligence and Statistics, pages 1540–1552. PMLR, 2020.

[9] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. arXiv preprint arXiv:2006.05929, 2020.

[10] Felipe Petroski Such, Aditya Rawal, Joel Lehman, Kenneth Stanley, and Jeffrey Clune. Generative teaching networks: Accelerating neural architecture search by learning to generate synthetic training data. In International Conference on Machine Learning, pages 9206–9216. PMLR, 2020.