Автоматическая регуляризация байесовских нейронных сетей

Препринт

Басов Дмитрий Константинович

Аннотация

В байесовском выводе, в отличие от гипотезы максимального правдоподобия, не делается никаких предположений о размере обучающей выборки. Это делает байесовские модели устойчивыми к переобучению.

Однако применение байесовского подхода сопряжено с такими проблемами, как задание подходящего априорного распределения и вычисление апостериорного распределения весов модели.

В данной работе предлагается следующее решение этих проблем:

1. Применяя вариационный вывод, мы аппроксимируем апостериорное распределение весов модели нормальным распределением с диагональной матрицей ковариации, и задача сводится к максимизации нижней вариационной границы. В этом случае каждый вес модели определяется двумя обучаемыми параметрами.
2. Априорное распределение весов модели задается в виде нормального распределения с нулевым математическим ожиданием и диагональной матрицей ковариации, элементы которой вычисляются из данных. Этот приём лежит в основе Relevance Vector Machine (RVM) — байесовского варианта Support Vector Machine (SVM).

Полученную модель можно рассматривать как ансамбль из бесконечного числа нейронных сетей, веса которых сэмплируются из нормального распределения. При этом каждый вес имеет свой индивидуальный коэффициент $L2$ регуляризации, который автоматически определяется из тренировочных данных при обучении.

Таким образом, предлагаемый подход обладает большой универсальностью и устойчивостью к переобучению.

1 Обозначения и сокращения

- $\boldsymbol{a} \odot \boldsymbol{b}$ — поэлементное произведение (произведение Адамара) векторов.
- $\boldsymbol{x}$ — вектор признаков.
- $\boldsymbol{y}$ — вектор целевой переменной.
- D — датасет — пары значений $\{\boldsymbol{x_i}, \boldsymbol{y_i}\}$, где $i = 1, \ldots, L$.
- $\boldsymbol{W}$ — веса модели — случайная величина размерности M.
- $p(\boldsymbol{y}|\boldsymbol{x}, D)$ — предсказательное распределение — вероятность (плотность вероятности) получить целевую переменную $\boldsymbol{y}$ для неразмеченного вектора $\boldsymbol{x}$, используя информацию из датасета D.
- $p(D|\boldsymbol{W}) = \prod_{i=1}^{L} p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{W})$ — правдоподобие (likelihood) — вероятность (плотность вероятности) получить датасет D при фиксированных весах модели $\boldsymbol{W}$.
- $p(\boldsymbol{W})$ — априорное распределение весов модели (prior). Посредством распределения $p(\boldsymbol{W})$ кодируется информация о весах модели до начала обучения.
- $p(D) = \int p(D|\boldsymbol{W})\, p(\boldsymbol{W})\, d\boldsymbol{W}$ — маргинальное правдоподобие (marginal likelihood).
- $p(\boldsymbol{W}|D)$ — апостериорное распределение весов модели (posterior). В данном распределении кодируется априорная информация о весах модели из распределения $p(\boldsymbol{W})$, дополненная информацией из датасета D посредством правдоподобия $p(D|\boldsymbol{W})$.

- $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ — аппроксимация апостериорного распределения весов модели $p(\boldsymbol{W}|D)$.
- $\boldsymbol{\theta}$ — обучаемые параметры байесовской модели. Стоит подчеркнуть, что в классическом машинном обучении веса и обучаемые параметры это одно и то же. Однако в байесовском подходе при использовании вариационного вывода обучаемыми параметрами являются параметры распределения $q_{\boldsymbol{\theta}}(\boldsymbol{W})$, из которого сэмплируются веса $\boldsymbol{W}$.
- $KL(q \mid\mid p) = \int q(\boldsymbol{Z}) \ln \dfrac{q(\boldsymbol{Z})}{p(\boldsymbol{Z})}\, d\boldsymbol{Z}$ — дивергенция Кульбака–Лейблера (KL–дивергенция). Если $q(\boldsymbol{Z}) = p(\boldsymbol{Z})$, то $KL(q \mid\mid p) = 0$. В противном случае $KL(q \mid\mid p) > 0$.
- $\mathcal{L}$ — Evidence Lower Bound (ELBO). Если $q_{\boldsymbol{\theta}}(\boldsymbol{W}) = p(\boldsymbol{W}|D)$, то $\mathcal{L} = p(D)$. В противном случае $\mathcal{L} < p(D)$.

2 Введение

В классическом машинном обучении делается следующее предположение: веса модели $\boldsymbol{W}$ являются пусть и неизвестной, но фиксированной величиной. В этом случае можно получить точечную оценку весов модели согласно гипотезе максимального правдоподобия:

$$\boldsymbol{W}^* = \underset{\boldsymbol{W}}{\operatorname{argmax}}\, p(D|\boldsymbol{W}) = \underset{\boldsymbol{W}}{\operatorname{argmax}} \sum_{i=1}^{L} \ln p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{W}) \tag{2.1}$$

Тогда предсказательное распределение $p(\boldsymbol{y}|\boldsymbol{x}, D)$ аппроксимируется следующим образом:

$$p(\boldsymbol{y}|\boldsymbol{x}, D) \approx p(\boldsymbol{y}|\boldsymbol{W}^*, \boldsymbol{x}) \tag{2.2}$$

Однако это справедливо при условии, что количество объектов в датасете D сильно больше количества весов модели ($L \gg M$). В противном случае веса модели $\boldsymbol{W}$ могут слишком сильно подстроиться под обучающую выборку D, что может привести к переобучению.

Для борьбы с переобучением используется ряд приёмов регуляризации (штрафы на норму весов, early stopping, dropout), однако для их настройки требуются дополнительные вычислительные ресурсы и отложенные (не участвующие в обучении) выборки данных.

В байесовском подходе вместо единственной модели с весами $\boldsymbol{W}^*$ используется усреднение ответов от всевозможных моделей таким образом, что вклад каждой модели пропорционален апостериорной вероятности $p(\boldsymbol{W}|D)$, которая определяется по теореме Байеса:

$$p(\boldsymbol{W}|D) = \frac{p(D|\boldsymbol{W})\, p(\boldsymbol{W})}{\int p(D|\boldsymbol{W})\, p(\boldsymbol{W})\, d\boldsymbol{W}} \tag{2.3}$$

Тогда предсказательное распределение $p(\boldsymbol{y}|\boldsymbol{x}, D)$ рассчитывается следующим образом:

$$p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{W})\, p(\boldsymbol{W}|D)\, d\boldsymbol{W} \tag{2.4}$$

Байесовские модели можно рассматривать как ансамбль из бесконечного числа моделей, веса которых сэмплируются из распределения $p(\boldsymbol{W}|D)$. Такой подход устойчив к переобучению, так как о размере обучающей выборки не делается никаких предположений. Однако возникают такие проблемы, как выбор подходящего априорного распределения $p(\boldsymbol{W})$ и вычисление апостериорного распределения $p(\boldsymbol{W}|D)$.

Неудачный выбор априорного распределения $p(\boldsymbol{W})$ может сильно снизить качество модели, а расчёт апостериорного распределения $p(\boldsymbol{W}|D)$ требует вычисления интеграла по всему пространству весов модели, что для нейронных сетей практически невозможно.

В данной статье предлагается следующий подход к решению этих проблем.

1. Применяя вариационный вывод, мы аппроксимируем распределение весов $p(\boldsymbol{W}|D)$ распределением $q_{\boldsymbol{\theta}}(\boldsymbol{W})$, и задача сводится к максимизации нижней вариационной границы $\mathcal{L}$ по параметрам $\boldsymbol{\theta}$.

2. Распределение $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ задаётся в виде нормального распределения с диагональной матрицей ковариации.
3. Так как распределение $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ является нормальным, применяя трюк с репараметризацией (reparameterization trick), становится возможным использовать градиентные методы для максимизации $\mathcal{L}$.
4. Априорное распределение весов модели $p(\boldsymbol{W})$ задаётся в виде нормального распределения с нулевым математическим ожиданием и диагональной матрицей ковариации, элементы которой определяются при обучении из датасета D. Такой подход обладает большой универсальностью, однако из-за определения параметров априорного распределения $p(\boldsymbol{W})$ из данных D теряется теоретическая устойчивость к переобучению. Идея определения некоторых параметров априорного распределения $p(\boldsymbol{W})$ из датасета D известна как эмпирический Байес.
5. Вводятся новые параметры $\boldsymbol{\gamma}$ и $\boldsymbol{\rho}$, через которые выражаются математическое ожидание и дисперсия распределения $q_{\boldsymbol{\theta}}(\boldsymbol{W})$. Это позволяет избежать неопределенности деления $\frac{0}{0}$, которая может возникнуть из–за определения дисперсии распределения $p(\boldsymbol{W})$ из данных.

3 Известные результаты

В [1, 2, 3] рассматривается применение байесовского подхода к нейронным сетям, в том числе использование эмпирического Байеса для нахождения гиперпараметров.

В работах [4, 5, 6] представлена техника Automatic Relevance Determination (ARD), которая легла в основу метода релевантых векторов (RVM) [7], байесовского варианта метода опорных векторов (SVM). В отличие от SVM, RVM не требует подбора коэффициента регуляризации.

В [1, 2, 3, 4, 5] при вычислении апостериорного распределения весов модели использовалась аппроксимация Лапласа, а в [6] для сэмплирования весов из апостериорного распределения применялись методы Монте–Карло для марковских цепей (MCMC). Недостатком данных подходов являются сложности с масштабированием на большие модели.

В работе [8] была применена техника вариационного вывода для аппроксимации апостериорного распределения весов модели нормальным распределением с диагональной матрицей ковариации. Параметры распределения были найдены аналитически, что возможно только в случае нейронных сетей с одним скрытым слоем.

В [9] для оценки параметров аппроксимации апостериорного распределения используются градиентные методы, что позволяет масштабировать вариационный вывод на большие модели.

В [10] для обучения байесовской нейронной сети используется трюк с репараметризацией, который до этого был реализован в модели вариационного автокодировщика [11].

Различные способы задания априоных распределений весов байесовских нейронных сетей рассматриваются в [12, 13, 14, 15].

Обзор байесовских методов машинного обучения приведён в книгах [16, 17, 18].

4 Постановка задачи

Задача машинного обучения с учителем в вероятностной постановке формулируется следующим образом: необходимо получить распределение вероятностей $p(\boldsymbol{y}|\boldsymbol{x}, D)$ целевой переменной $\boldsymbol{y}$ для неразмеченных $\boldsymbol{x}$, используя информацию из датасета D. В случае параметрических моделей, которыми являются нейронные сети, информация из датасета D кодируется посредством весов модели $\boldsymbol{W}$. Сделаем следующие преобразования:

$$p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}, \boldsymbol{W}|\boldsymbol{x}, D)\, d\boldsymbol{W} = \int p(\boldsymbol{y}|\boldsymbol{W}, \boldsymbol{x}, D)\, p(\boldsymbol{W}|\boldsymbol{x}, D)\, d\boldsymbol{W} = \int p(\boldsymbol{y}|\boldsymbol{W}, \boldsymbol{x})\, p(\boldsymbol{W}|D)\, d\boldsymbol{W} \tag{4.1}$$

Пояснения:

- $p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}, \boldsymbol{W}|\boldsymbol{x}, D)\, d\boldsymbol{W}$, так как для любых случайных величин $\boldsymbol{a}$ и $\boldsymbol{b}$ справедливо $p(\boldsymbol{a}) = \int p(\boldsymbol{a}, \boldsymbol{b})\, d\boldsymbol{b}$;

- $p(\boldsymbol{y}, \boldsymbol{W}|\boldsymbol{x}, D) = p(\boldsymbol{y}|\boldsymbol{W}, \boldsymbol{x}, D)\, p(\boldsymbol{W}|\boldsymbol{x}, D)$, так как для любых случайных величин $\boldsymbol{a}$ и $\boldsymbol{b}$ справедливо $p(\boldsymbol{a}, \boldsymbol{b}) = p(\boldsymbol{a}|\boldsymbol{b})\, p(\boldsymbol{b})$;
- $p(\boldsymbol{y}|\boldsymbol{W}, \boldsymbol{x}, D) = p(\boldsymbol{y}|\boldsymbol{W}, \boldsymbol{x})$, так как вся информация из датасета D отражена в весах $\boldsymbol{W}$;
- $p(\boldsymbol{W}|\boldsymbol{x}, D) = p(\boldsymbol{W}|D)$, так как веса модели $\boldsymbol{W}$ не зависят от неразмеченных $\boldsymbol{x}$, которых не было в датасете D.

Таким образом, для получения предсказательного распределения $p(\boldsymbol{y}|\boldsymbol{x}, D)$ необходимо получить распределение $p(\boldsymbol{W}|D)$. Получим выражение для $p(\boldsymbol{W}|D)$, используя формулу Байеса:

$$p(\boldsymbol{W}|D) = \frac{p(\boldsymbol{W}, D)}{p(D)} = \frac{p(\boldsymbol{W}, D)}{\int p(\boldsymbol{W}, D)\, d\boldsymbol{W}} = \frac{p(D|\boldsymbol{W})\, p(\boldsymbol{W})}{\int p(D|\boldsymbol{W})\, p(\boldsymbol{W})\, d\boldsymbol{W}} \tag{4.2}$$

Так как аналитическое вычисление интеграла в знаменателе (4.2) для нейронных сетей практически невозможно, возникает необходимость использовать различные аппроксимации для вычисления предсказательного распределения (4.1). Рассмотрим некоторые из них.

4.1 Точечная оценка весов модели

Вместо нахождения распределения весов $p(\boldsymbol{W}|D)$ можно сделать точечную оценку весов модели $\boldsymbol{W}$. Эту оценку можно получить путём максимизации плотности вероятности весов $p(\boldsymbol{W}|D)$:

$$\boldsymbol{W}^* = \operatorname*{argmax}_{\boldsymbol{W}} p(\boldsymbol{W}|D) = \operatorname*{argmax}_{\boldsymbol{W}} \ln\left(p(D|\boldsymbol{W}) \cdot p(\boldsymbol{W})\right) = \operatorname*{argmax}_{\boldsymbol{W}} \Big(\sum_{i=1}^{L} \ln p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{W}) + \ln p(\boldsymbol{W})\Big) \tag{4.3}$$

$\boldsymbol{W}^*$ называется максимальной апостериорной оценкой. Задание различных функциональных форм распределений $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{W})$ и $p(\boldsymbol{W})$ эквивалентно различным функциям потерь, которые используются в классическом машинном обучении. Рассмотрим некоторые примеры:

- задание $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{W})$ в виде распределения Бернулли равнозначно использованию логистической функции потерь;
- задание $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{W})$ в виде нормального распределения равнозначно использованию метода наименьших квадратов;
- задание $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{W})$ в виде распределения Лапласа равнозначно использованию метода наименьших модулей;
- задание $p(\boldsymbol{W})$ в виде нормального распределения с нулевым математическим ожиданием равнозначно использованию L2 регуляризации, где коэффициент регуляризации определяется дисперсией априорного распределения;
- задание $p(\boldsymbol{W})$ в виде распределения Лапласа с нулевым параметром сдвига равнозначно использованию L1 регуляризации, где коэффициент регуляризации определяется параметром масштаба априорного распределения;
- задание $p(\boldsymbol{W})$ в виде равномерного распределения равнозначно отсутствию регуляризации, и максимальная апостериорная оценка (4.3) превращается в оценку максимального правдоподобия (2.1).

Тогда предсказательное распределение (4.1) аппроксимируется следующим образом:

$$p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}|\boldsymbol{W}, \boldsymbol{x})\, p(\boldsymbol{W}|D)\, d\boldsymbol{W} \approx p(\boldsymbol{y}|\boldsymbol{W}^*, \boldsymbol{x}) \tag{4.4}$$

Обучаемыми параметрами являются веса модели $\boldsymbol{W}^*$. Недостатком подхода точечной оценки весов модели является неустойчивость к переобучению. Для борьбы с переобучением используется множество приёмов, для настройки которых требуются дополнительные вычислительные ресурсы и валидационные (не участвующие в обучении) выборки данных.

4.2 MCMC

Для получения предсказательного распределения $p(\boldsymbol{y}|\boldsymbol{x}, D)$ можно воспользоваться следующим подходом: используя методы Монте-Карло для марковских цепей (MCMC), из распределения $p(\boldsymbol{W}|D)$ сэмплируются веса $\boldsymbol{W}$. Тогда предсказательное распределение (4.1) можно аппроксимировать следующим образом:

$$p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}|\boldsymbol{W}, \boldsymbol{x})\, p(\boldsymbol{W}|D)\, d\boldsymbol{W} \approx \frac{1}{T} \sum_{t=1}^{T} p(\boldsymbol{y}|\boldsymbol{x}, \hat{\boldsymbol{W}}_{\boldsymbol{t}}) \tag{4.5}$$

где $\hat{\boldsymbol{W}}_{\boldsymbol{t}}$ — сэмпл весов модели из распределения $p(\boldsymbol{W}|D)$ и T — количество сэмплов для аппроксимации. Чем больше T, тем выше точность аппроксимации.

Также стоит заметить, что в этом случае нет ни обучаемых параметров, ни функции потерь, которые присущи классическому машинному обучению.

Однако сэмплирование весов из распределения $p(\boldsymbol{W}|D)$ с использованием методов MCMC требует большого количества вычислительных ресурсов, что делает такой подход практически неприменимым для больших датасетов и большого числа весов.

4.3 Аппроксимация апостериорного распределения

Так как сэмплировать веса из распределения $p(\boldsymbol{W}|D)$ вычислительно сложно, можно аппроксимировать апостериорное распределение $p(\boldsymbol{W}|D)$ распределением $q_{\boldsymbol{\theta}}(\boldsymbol{W})$, из которого сэмплировать намного проще.

Тогда предсказательное распределение (4.1) аппроксимируется следующим образом:

$$p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}|\boldsymbol{W}, \boldsymbol{x})\, p(\boldsymbol{W}|D)\, d\boldsymbol{W} \approx \int p(\boldsymbol{y}|\boldsymbol{W}, \boldsymbol{x})\, q_{\boldsymbol{\theta}}(\boldsymbol{W})\, d\boldsymbol{W} \approx \frac{1}{T} \sum_{t=1}^{T} p(\boldsymbol{y}|\boldsymbol{x}, \hat{\boldsymbol{W}}_{\boldsymbol{t}}) \tag{4.6}$$

где $\hat{\boldsymbol{W}}_{\boldsymbol{t}}$ — сэмпл весов модели из распределения $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ и T — количество сэмплов для аппроксимации.

В этом случае обучаемыми параметрами являются $\boldsymbol{\theta}$ — параметры распределения $q_{\boldsymbol{\theta}}(\boldsymbol{W})$, из которого сэмплируются веса $\boldsymbol{W}$. Одним из подходов к аппроксимации сложного распределения более простым распределением является вариационный вывод.

5 Вариационный вывод

Идея вариационного вывода — сведение задачи байесовского вывода к задаче максимизации нижней вариационной границы (ELBO) $\mathcal{L}$, которая для распределения $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ записывается следующим образом:

$$\mathcal{L} = \int q_{\boldsymbol{\theta}}(\boldsymbol{W})\, \ln \frac{p(\boldsymbol{W}, D)}{q_{\boldsymbol{\theta}}(\boldsymbol{W})}\, d\boldsymbol{W} \tag{5.1}$$

Покажем мотивацию максимизации $\mathcal{L}$. Запишем выражение для $KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \parallel p(\boldsymbol{W}|D))$ и преобразуем его, используя тождество $p(\boldsymbol{W}, D) = p(\boldsymbol{W}|D)\, p(D)$:

$$\begin{aligned} KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \parallel p(\boldsymbol{W}|D)) = \\ = \int q_{\boldsymbol{\theta}}(\boldsymbol{W})\, \ln \frac{q_{\boldsymbol{\theta}}(\boldsymbol{W})}{p(\boldsymbol{W}|D)} d\boldsymbol{W} = \\ = \int q_{\boldsymbol{\theta}}(\boldsymbol{W})\, \ln \frac{p(D)\, q_{\boldsymbol{\theta}}(\boldsymbol{W})}{p(\boldsymbol{W}, D)} d\boldsymbol{W} = \\ = \ln p(D) \int q_{\boldsymbol{\theta}}(\boldsymbol{W}) d\boldsymbol{W} - \int q_{\boldsymbol{\theta}}(\boldsymbol{W})\, \ln \frac{p(\boldsymbol{W}, D)}{q_{\boldsymbol{\theta}}(\boldsymbol{W})} d\boldsymbol{W} = \\ = \ln p(D) - \mathcal{L} \end{aligned} \tag{5.2}$$

Так как $\ln p(D)$ не зависит от $\boldsymbol{\theta}$, максимизация $\mathcal{L}$ по параметрам $\boldsymbol{\theta}$ ведёт к минимизации $KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \,||\, p(\boldsymbol{W}|D))$. Таким образом, при максимизации $\mathcal{L}$ распределение $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ будет приближаться к распределению $p(\boldsymbol{W}|D)$.

Преобразуем выражение для $\mathcal{L}$, используя тождество $p(\boldsymbol{W}, D) = p(D|\boldsymbol{W})\, p(\boldsymbol{W})$:

$$\begin{aligned} \mathcal{L} = \\ = \int q_{\boldsymbol{\theta}}(\boldsymbol{W})\, \ln \frac{p(\boldsymbol{W}, D)}{q_{\boldsymbol{\theta}}(\boldsymbol{W})} d\boldsymbol{W} = \\ = \int q_{\boldsymbol{\theta}}(\boldsymbol{W})\, \ln \frac{p(D|\boldsymbol{W})\, p(\boldsymbol{W})}{q_{\boldsymbol{\theta}}(\boldsymbol{W})} d\boldsymbol{W} = \\ = \int q_{\boldsymbol{\theta}}(\boldsymbol{W})\, \ln p(D|\boldsymbol{W}) d\boldsymbol{W} - \int q_{\boldsymbol{\theta}}(\boldsymbol{W})\, \ln \frac{q_{\boldsymbol{\theta}}(\boldsymbol{W})}{p(\boldsymbol{W})} d\boldsymbol{W} = \\ = \int q_{\boldsymbol{\theta}}(\boldsymbol{W})\, \ln p(D|\boldsymbol{W}) d\boldsymbol{W} - KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \,||\, p(\boldsymbol{W})) \end{aligned} \tag{5.3}$$

Поставив $p(D|\boldsymbol{W}) = \prod_{i=1}^{L} p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{W})$ в формулу (5.3), получим:

$$\mathcal{L} = \int q_{\boldsymbol{\theta}}(\boldsymbol{W}) \sum_{i=1}^{L} \ln p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{W}) d\boldsymbol{W} - KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \,||\, p(\boldsymbol{W})) \tag{5.4}$$

Таким образом, выражение для ELBO раскладывается на две составляющие. Левая часть показывает, насколько хорошо распределение весов модели $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ описывает датасет D. Правая часть, которая является регуляризацией, показывает, насколько сильно распределение весов $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ отличается от априорного распределения весов $p(\boldsymbol{W})$.

6 Задание функциональных форм распределений

Для дальнейшнего вывода положим, что распределения $p(\boldsymbol{W})$ и $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ являются нормальными с диагональными матрицами ковариации:

$$q_{\boldsymbol{\theta}}(\boldsymbol{W}) = N(\boldsymbol{W}|\boldsymbol{\mu}, diag(\boldsymbol{\sigma})^2) = \prod_{k=1}^{M} N(W_k|\mu_k, \sigma_k^2) \tag{6.1}$$

$$p(\boldsymbol{W}) = N(\boldsymbol{W}|\boldsymbol{0}, diag(\boldsymbol{\delta})^2) = \prod_{k=1}^{M} N(W_k|0, \delta_k^2) \tag{6.2}$$

Так как распределения $p(\boldsymbol{W})$ и $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ нормальные, то $KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \,||\, p(\boldsymbol{W}))$ можно посчитать аналитически:

$$KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \,||\, p(\boldsymbol{W})) = \frac{1}{2} \sum_{k=1}^{M} (\frac{\sigma_k^2}{\delta_k^2} + \frac{\mu_k^2}{\delta_k^2} - \ln \frac{\sigma_k^2}{\delta_k^2} - 1) \tag{6.3}$$

Подставив (6.3) в (5.4), получим:

$$\mathcal{L} = \int q_{\boldsymbol{\theta}}(\boldsymbol{W}) \sum_{i=1}^{L} \ln p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{W}) d\boldsymbol{W} - \frac{1}{2} \sum_{k=1}^{M} (\frac{\sigma_k^2}{\delta_k^2} + \frac{\mu_k^2}{\delta_k^2} - \ln \frac{\sigma_k^2}{\delta_k^2} - 1) \tag{6.4}$$

Таким образом, $\boldsymbol{\mu}$ и $\boldsymbol{\sigma}$ — это обучаемые параметры модели, а параметр $\boldsymbol{\delta}$ — гиперпараметр (так как является параметром априорного распределения $p(\boldsymbol{W})$). В байесовском выводе все параметры априорного распределения должны задаваться до начала обучения. Однако мы можем определить параметр $\boldsymbol{\delta}$ из данных. Такой прием, как уже указывалось, называется эмпирический Байес.

7 Эмпирический Байес

Найдём такое значение $\boldsymbol{\delta}$, при котором $\mathcal{L}$ максимальна. Так как в выражении (5.4) левое слагаемое не зависит от параметров распределения $p(\boldsymbol{W})$, то максимум $\mathcal{L}$ достигается при минимуме $KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \,||\, p(\boldsymbol{W}))$ по параметру $\boldsymbol{\delta}$.

Пусть $\boldsymbol{\alpha} = diag(\boldsymbol{\delta})^{-2}$. Тогда выражение (6.3) будет иметь следующий вид:

$$KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \mid\mid p(\boldsymbol{W})) = \frac{1}{2}\sum_{k=1}^{M}\left(\alpha_k \cdot \sigma_k^2 + \alpha_k \cdot \mu_k^2 - (\ln\sigma_k^2 + \ln\alpha_k) - 1\right) \tag{7.1}$$

Найдём производную $KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \mid\mid p(\boldsymbol{W}))$ по параметру α_k:

$$\frac{\partial(KL(q_{\boldsymbol{\theta}}(\boldsymbol{W}) \mid\mid p(\boldsymbol{W})))}{\partial\alpha_k} = \frac{1}{2}\left(\sigma_k^2 + \mu_k^2 - \frac{1}{\alpha_k}\right) = \frac{1}{2}\left(\sigma_k^2 + \mu_k^2 - \delta_k^2\right) \tag{7.2}$$

Приравняв (7.2) к нулю, получим выражение для оптимального значения $\boldsymbol{\delta}$:

$$\delta_k^2 = \sigma_k^2 + \mu_k^2 \tag{7.3}$$

Подставив (7.3) в (6.4), получим:

$$\mathcal{L} = \int q_{\boldsymbol{\theta}}(\boldsymbol{W})\sum_{i=1}^{L}\ln p(\boldsymbol{y_i}|\boldsymbol{x_i}, \boldsymbol{W})d\boldsymbol{W} - \frac{1}{2}\sum_{k=1}^{M}\ln\left(1 + \frac{\mu_k^2}{\sigma_k^2}\right) \tag{7.4}$$

Таким образом, задача свелась к максимизации $\mathcal{L}$ по параметрам $\boldsymbol{\mu}$ и $\boldsymbol{\sigma}$. Покажем, как свести задачу максимизации $\mathcal{L}$ к задаче градиентной оптимизации.

8 Градиентная оптимизация

Так как в случае нейронной сети аналитически посчитать интеграл по всему пространству весов $\boldsymbol{W}$ в (7.4) не представляется возможным, воспользуемся следующей аппроксимацией для $\mathcal{L}$:

$$\mathcal{L} \approx \frac{1}{S}\sum_{j=1}^{S}\sum_{i=1}^{L}\ln p(\boldsymbol{y_i}|\boldsymbol{x_i}, \hat{\boldsymbol{W}}_{\boldsymbol{ij}}) - \frac{1}{2}\sum_{k=1}^{M}\ln\left(1 + \frac{\mu_k^2}{\sigma_k^2}\right) \tag{8.1}$$

где $\hat{\boldsymbol{W}}_{\boldsymbol{ij}}$ — сэмпл весов модели из распределения $q_{\boldsymbol{\theta}}(\boldsymbol{W})$, S — количество сэмлов для аппроксимации, L — количество объектов в датасете D.

Так как распределение $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ нормальное, мы можем использовать трюк с репараметризацией при сэмплировании весов, что позволяет применять градиентные методы для максимизации $\mathcal{L}$ (8.1) по параметрам $\boldsymbol{\sigma}$ и $\boldsymbol{\mu}$:

$$\hat{\boldsymbol{W}}_{\boldsymbol{ij}} = \boldsymbol{\varepsilon_{ij}} \odot \boldsymbol{\sigma} + \boldsymbol{\mu} \tag{8.2}$$

где $\boldsymbol{\varepsilon_{ij}} \sim N(\boldsymbol{0}, \boldsymbol{I})$ — сэмплы из стандартного нормального распределения.

Задача максимизации (8.1) свелась к градиентной оптимизации. Однако градиентная оптимизация $\mathcal{L}$ по параметрам $\boldsymbol{\mu}$ и $\boldsymbol{\sigma}$ может привести к численной нестабильности. Эта проблема решается заменой переменных.

9 Замена переменных

В некоторых случаях при максимизации $\mathcal{L}$ какой-либо вес модели может перестать быть случайной величиной и выродиться в ноль ($\sigma_k \to 0$ и $\mu_k \to 0$). Это приведет к неопределенности деления 0 на 0 при вычислении KL–дивергенции в (8.1).

Также при градиентной оптимизации (8.1) компоненты вектора $\boldsymbol{\sigma}$ могут попасть в отрицательную область, что нежелательно, так как среднеквадратическое отклонение не может быть отрицательным по определению.

Чтобы избежать этих проблем, определим параметры $\boldsymbol{\sigma}$ и $\boldsymbol{\mu}$ через новые параметры $\boldsymbol{\rho}$ и $\boldsymbol{\gamma}$ следующим образом:

$$\boldsymbol{\sigma} = \ln(1 + e^{\boldsymbol{\rho}}) = Softplus(\boldsymbol{\rho}) \tag{9.1}$$

$$\boldsymbol{\mu} = \boldsymbol{\gamma} \odot \boldsymbol{\sigma} = \boldsymbol{\gamma} \odot Softplus(\boldsymbol{\rho}) \tag{9.2}$$

Подставив (9.1) и (9.2) в (8.1), получим:

$$\mathcal{L} \approx \frac{1}{S} \sum_{j=1}^{S} \sum_{i=1}^{L} \ln p(\boldsymbol{y_i}|\boldsymbol{x_i}, \hat{\boldsymbol{W}}_{\boldsymbol{ij}}) - \frac{1}{2} \sum_{k=1}^{M} \ln(1 + \gamma_k^2) \tag{9.3}$$

Таким образом, задача свелась к максимизации (9.3) по параметрам $\boldsymbol{\rho}$ и $\boldsymbol{\gamma}$. Значение $\hat{\boldsymbol{W}}_{\boldsymbol{ij}}$ вычисляется по (8.2), которое, в свою очередь, вычисляется через (9.1) и (9.2).

10 Алгоритм обучения

Возьмем выражение для $\mathcal{L}$ из (9.3), и разделив на размер обучеющей выборки L со знаком минус, получим следующую функцию потерь:

$$loss(\boldsymbol{\rho}, \boldsymbol{\gamma}) = -\frac{1}{S \cdot L} \sum_{j=1}^{S} \sum_{i=1}^{L} \ln p(\boldsymbol{y_i}|\boldsymbol{x_i}, \hat{\boldsymbol{W}}_{\boldsymbol{ij}}) + \frac{\frac{1}{2} \sum_{k=1}^{M} \ln(1 + \gamma_k^2)}{L} \tag{10.1}$$

Для ускорения расчёта функции потерь на каждом градиентном шаге введём следующие упрощения:

- на каждый объект будем делать только один сэмпл весов (то есть задать $S = 1$);
- средний отрицательный логарифм правдоподобия будем считать не по всей обучающих выборке, а на случайном подмножестве (батче).

Тогда выражение (10.1) для функции потерь будет выглядить следующим образом:

$$loss(\boldsymbol{\rho}, \boldsymbol{\gamma}) \approx -\frac{1}{B} \sum_{b=1}^{B} \ln p(\boldsymbol{y_b}|\boldsymbol{x_b}, \hat{\boldsymbol{W}}_{\boldsymbol{b}}) + \frac{\frac{1}{2} \sum_{k=1}^{M} \ln(1 + \gamma_k^2)}{L} \tag{10.2}$$

где:

- $\boldsymbol{\rho}$ и $\boldsymbol{\gamma}$ — параметры распределения $q_{\boldsymbol{\theta}}(\boldsymbol{W})$ (обучаемые параметры модели);
- $\hat{\boldsymbol{W}}_{\boldsymbol{b}}$ — сэмпл весов модели из распределения $q_{\boldsymbol{\theta}}(\boldsymbol{W})$;
- B — количество объектов в батче;
- L — количество объектов в датасете D;
- M — количество весов в модели.

Для минимизации (10.2) можно использовать любые методы градиентной оптимизации. Для наглядности рассмотрим алгоритм стохастического градиентного спуска.

Задаем шаг градиентного спуска η и инициализируем обучаемые параметры $\boldsymbol{\rho}$ и $\boldsymbol{\gamma}$. Затем повторяем, пока не достигнем критерия остановки:

1. $\boldsymbol{\sigma} \leftarrow Softplus(\boldsymbol{\rho})$ — расчёт среднеквадратических отклонений весов
2. $\boldsymbol{\mu} \leftarrow \boldsymbol{\gamma} \odot \boldsymbol{\sigma}$ — расчёт математических ожиданий весов
3. $\boldsymbol{\varepsilon} \leftarrow N(0, 1)$ — сэмплирование стандартного нормального шума
4. $\hat{\boldsymbol{W}} \leftarrow \boldsymbol{\varepsilon} \odot \boldsymbol{\sigma} + \boldsymbol{\mu}$ — репараметризация
5. $nll \leftarrow -\frac{1}{B} \sum_{b=1}^{B} \ln p(\boldsymbol{y_b}|\boldsymbol{x_b}, \hat{\boldsymbol{W}})$ — расчёт среднего отрицательного логарифма правдоподобия
6. $kl \leftarrow \frac{1}{2} \sum_{k=1}^{M} \ln(1 + \gamma_k^2)$ — расчёт KL–дивергенции
7. $l \leftarrow nll + \frac{kl}{L}$ — расчёт функции потерь
8. $\boldsymbol{\rho} \leftarrow \boldsymbol{\rho} - \eta \frac{\partial l}{\partial \boldsymbol{\rho}}$ — обновление $\boldsymbol{\rho}$
9. $\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma} - \eta \frac{\partial l}{\partial \boldsymbol{\gamma}}$ — обновление $\boldsymbol{\gamma}$.

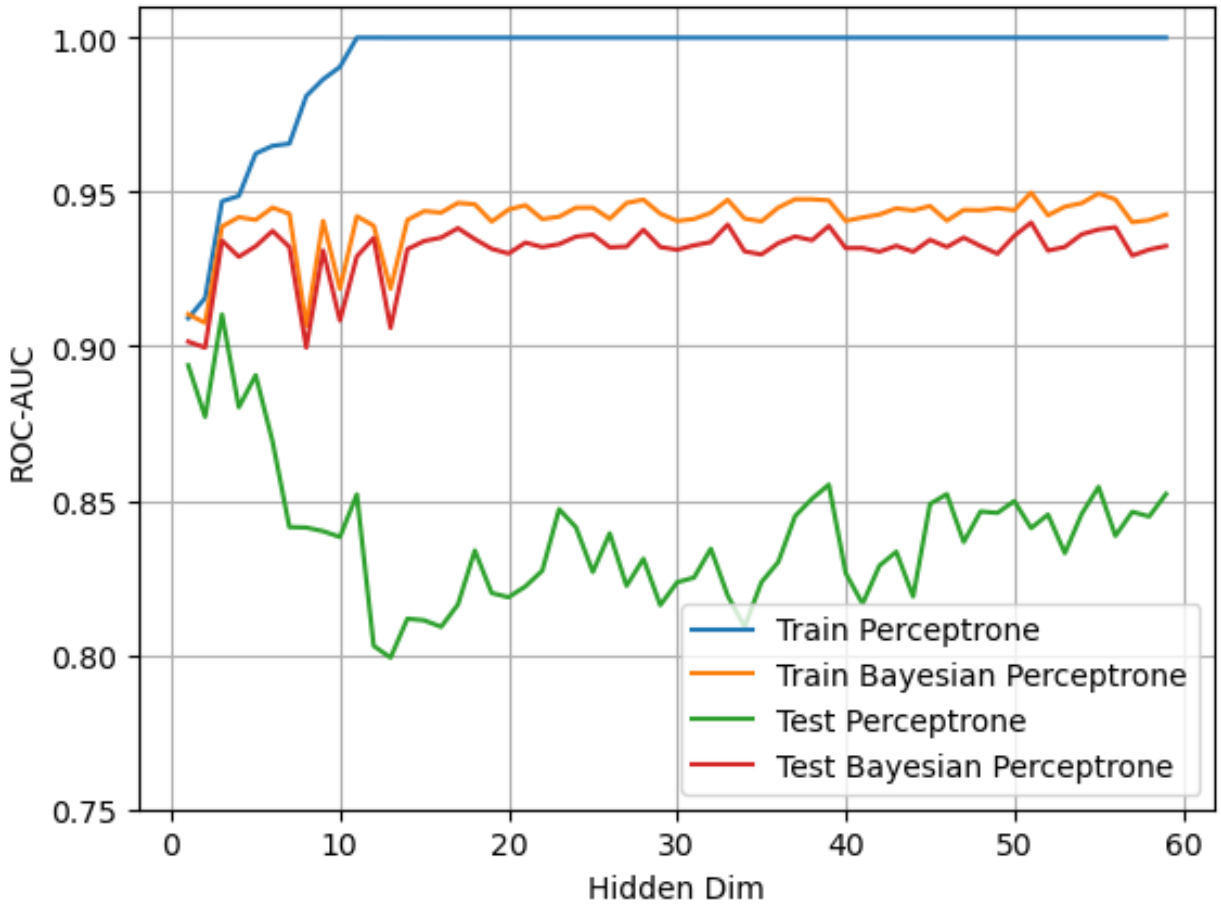

Рис. 1: Зависимость ROC–AUC от размерности скрытого состояния на тренировочных и тестовых данных

11 Эксперименты

Для проверки гипотезы был выбран Alzheimer's Disease Dataset. Данные были разбиты на тренировочную и тестовую часть в пропорции 80 на 20. В качестве архитектуры была выбрана полносвязная нейронная сеть с одним скрытым слоем и функцией активации ReLU, то есть:
$z = ReLU(matmul(x, W_1))$
$y = Sigmoid(matmul(z, W_2))$.

Размерность скрытого состояния z варьировалась от 1 до 60. Для каждой размерности обучались 2 модели: классическая нейронная сеть и байесовская нейронная сеть. Для каждой модели производилась оценка ROC–AUC на тренировочной и тестовой выборках.

Цель экспериментов: показать, что с увеличением сложности модели байесовская нейронная сеть не переобучается.

На рисунке 1 представлены результаты экспериментов.

12 Выводы

По результатам работы можно сделать следующие выводы:

- с ростом сложности модели байесовская нейронная сеть не переобучилась;

- значение ROC-AUC на тестовой выборке имеет очень высокую корреляцию со значением ROC-AUC на тренировочной выборке (0.97 по Пирсону). Следовательно, для подбора гиперпараметров можно ориентироваться на метрики, полученные на тренировочной выборке. Это даёт возможность отказаться от деления на тренировочную и валидационную выборки для подбора гиперпараметров.

Также стоит отметить, что данный подход переносится на другие архитектуры нейронных сетей (рекуррентные, свёрточные, трансформеры).

Имплементация данного подхода была выполнена с использованием PyTorch. Весь исходный код для проведения экспериментов размещён по адресу `https://github.com/dimabasow/bayesian-neural-networks`.

Список литературы

[1] MacKay, D. J. C. (1992a). Bayesian interpolation. Neural Computation 4(3), 415–447.

[2] MacKay, D. J. C. (1992b). The evidence framework applied to classification networks. Neural Computation 4(5), 720–736.

[3] MacKay, D. J. C. (1992c). A practical Bayesian framework for back-propagation networks. Neural Computation 4(3), 448–472.

[4] MacKay, D. J. C. (1994). Bayesian methods for backprop networks. In E. Domany, J. L. van Hemmen, and K. Schulten (Eds.), Models of Neural Networks, III, Chapter 6, pp. 211–254. Springer.

[5] MacKay, D. J. C. (1995). Probable networks and plausible predictions — a review of practical Bayesian methods for supervised neural networks". In: Network: Computation in Neural Systems 6.3, pp. 469–505.

[6] R. Neal. (1996). Bayesian learning for neural networks. Springer.

[7] Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine. Journal of Machine Learning Research 1, 211–244.

[8] Hinton, G. E. and D. van Camp (1993). Keeping neural networks simple by minimizing the description length of the weights. In Proceedings of the Sixth Annual Conference on Computational Learning Theory, pp. 5–13. ACM.

[9] A. Graves (2011). Practical Variational Inference for Neural Networks. NIPS.

[10] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. (2015). Weight Uncertainty in Neural Networks. ICML.

[11] Kingma D. P., Welling M. (2013). Welling M. Auto-Encoding Variational Bayes. ArXiv e-prints.

[12] E. T. Nalisnick. (2018). On Priors for Bayesian Neural Networks. PhD thesis. UC Irvine.

[13] A. G. Wilson and P. Izmailov. (2020). Bayesian Deep Learning and a Probabilistic Perspective of Generalization. NIPS.

[14] T. Hoefler, D. Alistarh, T. Ben-Nun, N. Dryden, and A. Peste. (2021). Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks. In arXiv: 2102.00554.

[15] V. Fortuin. (2022). Priors in Bayesian Deep Learning: A Review. In: Intl. Statisical Review.

[16] C. Bishop (2006). Pattern recognition and machine learning. Springer.

[17] Kevin P. Murphy. (2022). Probabilistic Machine Learning: An introduction. MIT Press.

[18] Kevin P. Murphy. (2023). Probabilistic Machine Learning: Advanced Topics. MIT Press.