Обучение бинарных нейросетевых моделей с помощью непрерывных аппроксимаций

Павлюченков Д.М.

2023 год

План презентации

- Введение
- Подходы к обучению бинарных сетей
- Постановка задачи
- Обучение бинарных нейросетевых моделей с помощью непрерывных аппроксимаций
- Эксперименты
- Заключение

Введение

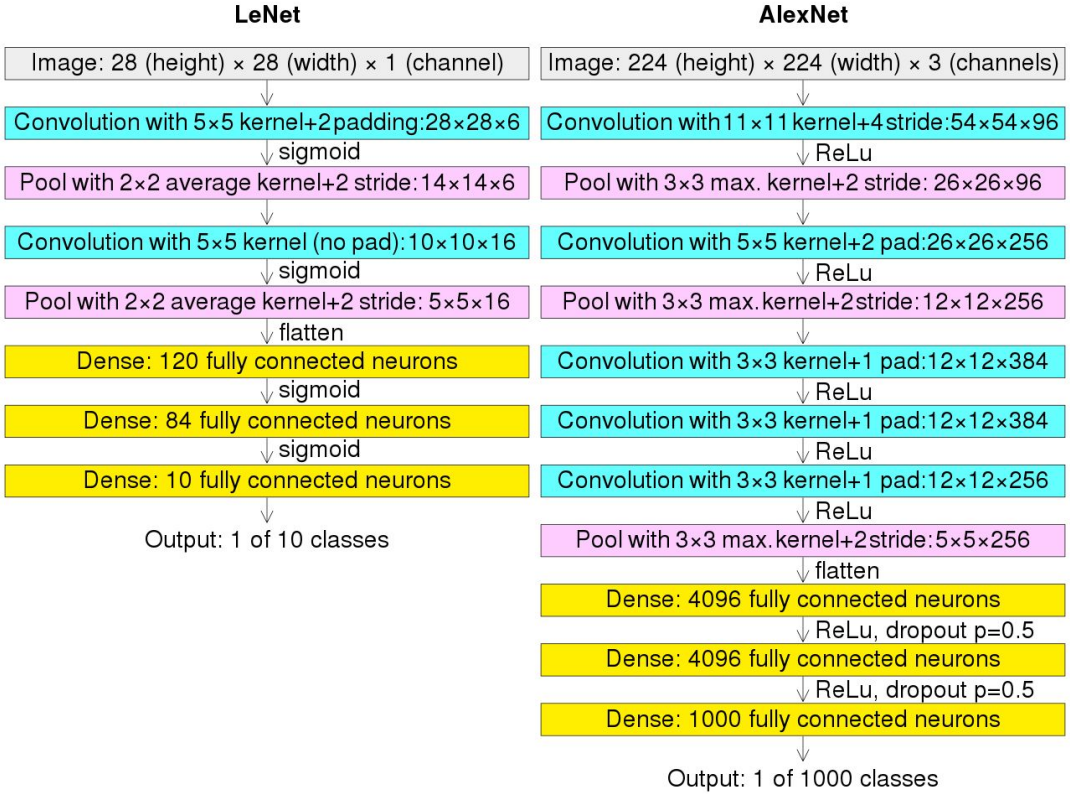
Сверточные сети и распознавание изображений

Напомню идеологию:

Сверточный слой – выделение “карты признаков” a.k.a. переход от конкретных особенностей изображения к понятиям высокого уровня

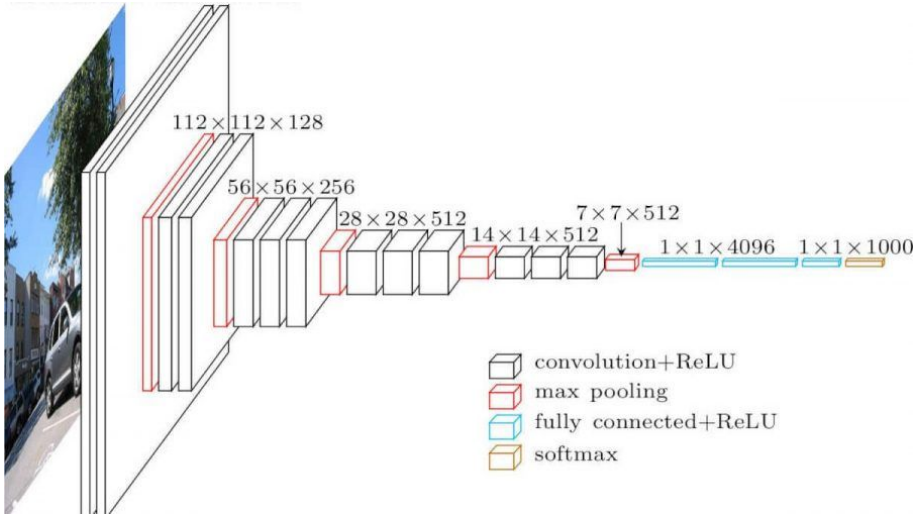
Функция активации – добавление нелинейности и отсечение отрицательной части скалярной величины

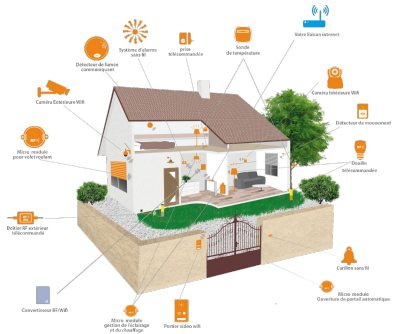
Пулинг – уплотнение карты признаков a.k.a. Переход на следующий уровень абстракции



Проблема

У современных сверточных сетей – миллионы параметров, поэтому они требуют больших ресурсов для хранения и работы.
Из-за этого мы не можем применять большие модели на маленьких устройствах.





Решение : квантование (бинаризация)

1. Весов и входного вектора:

-1/12 3.3 1000.7

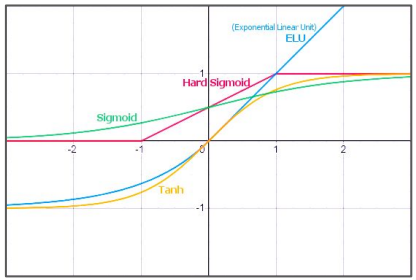
42 -14.88

→

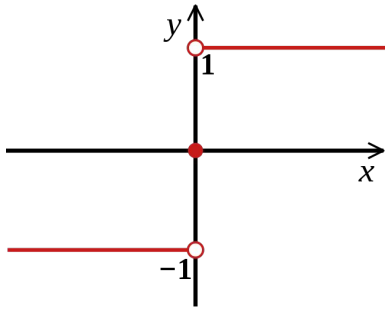
-1 1 1

1 -1

2. Функций активации:

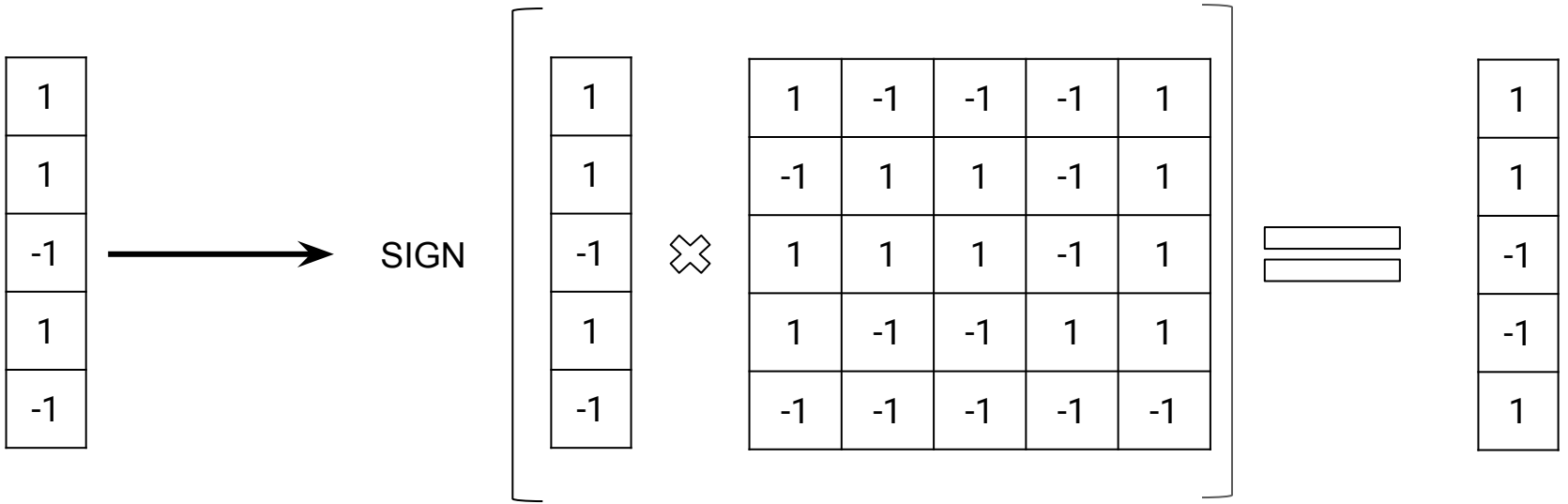


→



Классический полносвязный слой бинарной сети

Рассмотрим классический полносвязный слой бинарной сети. Ему на вход поступает вектор размера N, далее он умножается на бинарную (состоящую из 1 и -1) матрицу весов W, и к произведению применяется пороговая функция активации. В результате получается бинарный вектор.



Бинарные нейронные сети : плюсы

1. В лучшем случае – в **32** раза меньше памяти на хранение по сравнению с вещественными сетями
2. Дорогая операция вещественного умножения заменяется на простые операции целочисленного сложения и смены знака
3. Возможно использовать битовые операции, что еще больше снижает нагрузку на процессор

Можно использовать сложные сети на маломощных устройствах

Бинарные нейронные сети : минусы

1. “Лучший случай” достигается не всегда, приходится увеличивать размер сети для сохранения качества

2. Стандартные методы обучения не подходят:
   - **sign – недифференцируемая функция**
   - **шаг между весами – дискретный**

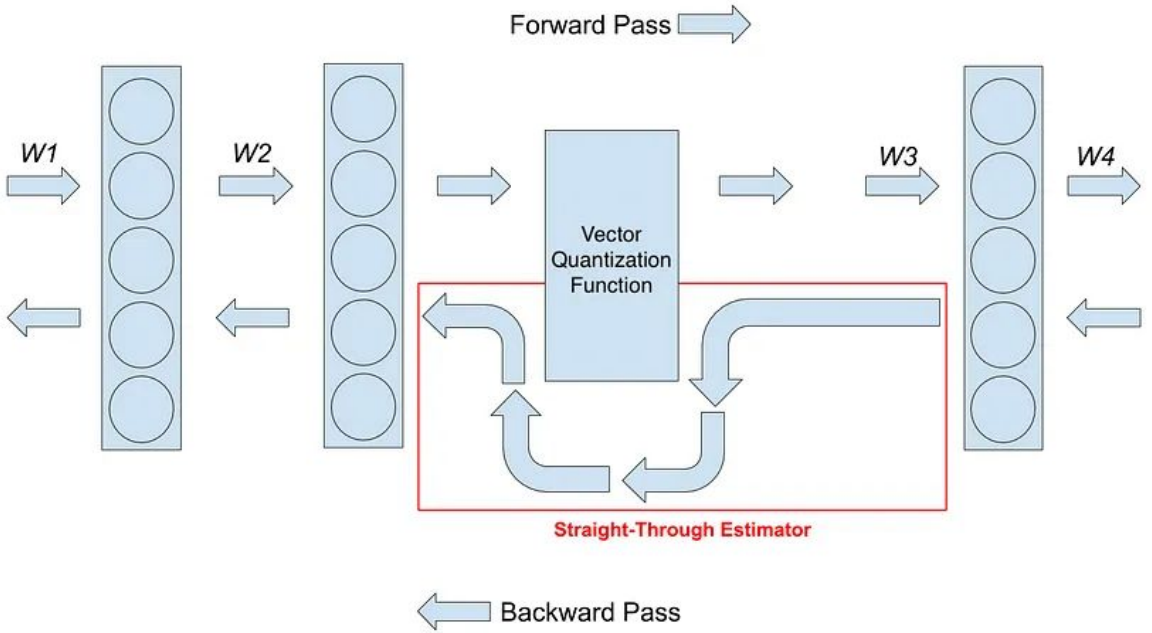
3. Универсальных методов обучения пока нет, то есть к каждой задаче методы нужно подбирать свои

Разработка эффективных методов обучения бинарных сетей представляет значительный научный и практический интерес

Подходы к обучению бинарных сетей

Подход пробрасывания градиента (STE)

STE (Straight-Through Estimator) - это метод обучения, который позволяет применять функции активации с недифференцируемыми точками. В этом методе при проходе обратно по сети градиент просто передается дальше без изменения.
Этот метод в работе используется для построения референсной модели.

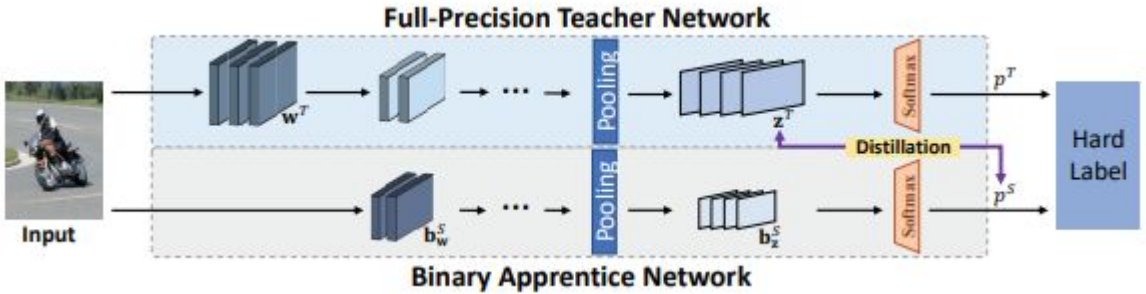


Подходы к обучению бинарных сетей

Обучение бинарных весов параллельно с вещественными

Вероятностные подходы

Параллельное обучение бинарной и вещественной сетей. На основании классической вещественной сети строится аналогичная сеть, но с уже бинаризованными весами и активациями. Затем по выходному результату осуществляется корректировка вещественных весов



Обучение бинарных весов параллельно с вещественными

Прямой проход:
Квантуем веса и активации, остальное как обычно
Обратный проход:
Вычисляем полноразрядные градиенты, используя градиенты бинарных весов и активаций. Затем веса и параметры обновляются.

Модификации:

1. Вместо sign – sigmoid с гиперпараметром
2. Регуляризация с гиперпараметром
3. New_loss = loss + k * regularization, где k – обучаемый

Рассмотрим семейство алгоритмов обучения BNN, которые позволяют параллельно обучать вещественные и бинарные веса. Этот подход был предложен Hubara et al. в [3], а одна из его модификаций описана в статье [4].Для прямого прохода авторы предлагают квантовать веса и активации. А в обратном проходе вычисляются полноразрядные градиенты, используя градиенты бинарных весов и активаций. Затем веса и параметры обновляются. Darabi et al. [4] рассмотрели три улучшения этого подхода. Во-первых, вместо использования функции знака в качестве функции активации, они предлагают аппроксимировать её с помощью сигмоидальной функции и дополнительного гиперпараметра. Это позволяет настраивать наилучшую аппроксимацию функции знака, которая при этом является дифференцируемой. Во-вторых, они вводят дополнительный параметр в функцию регуляризации, который также можно обучать. Этот параметр вычисляется динамически в процессе обучения через статистики весов. В классическом случае цель регуляризации заключается в предотвращении переобучения, например, через уменьшение модулей весов. Однако в этом случае регуляризация используется для приближения вещественных весов к бинарным значениям -1 или +1 и штрафу за удаление от них. В-третьих, с помощью регуляризации модифицируют функцию потерь для бинарного случая и делают её обучаемой. К классической функции потерь авторы прибавляют сумму функций регуляризации по слоям, домноженную на параметр, который также можно настраивать и обучать для повышения точности распознавания.

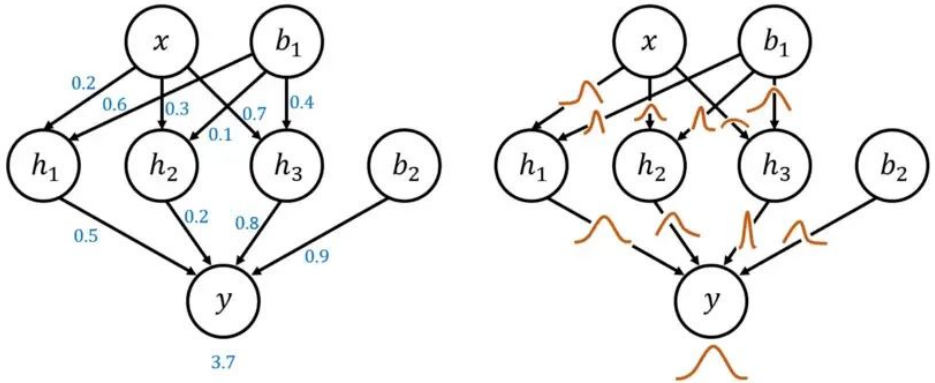
Осуществляется параллельное обучение бинарной и веществennозначной сетей. На основании классической вещественной сети строится аналогичная сеть, но с уже бинаризованными весами и активациями. Затем по выходному результату осуществляется корректировка вещественных весов.

Подходы к обучению бинарных сетей

Обучение бинарных весов параллельно с вещественными

Вероятностные подходы

В рамках вероятностных подходов обучаются стохастические нейронные сети (СНС), в которых для коэффициентов и активаций обучаются не одно конкретное значение, а *распределение* значений.

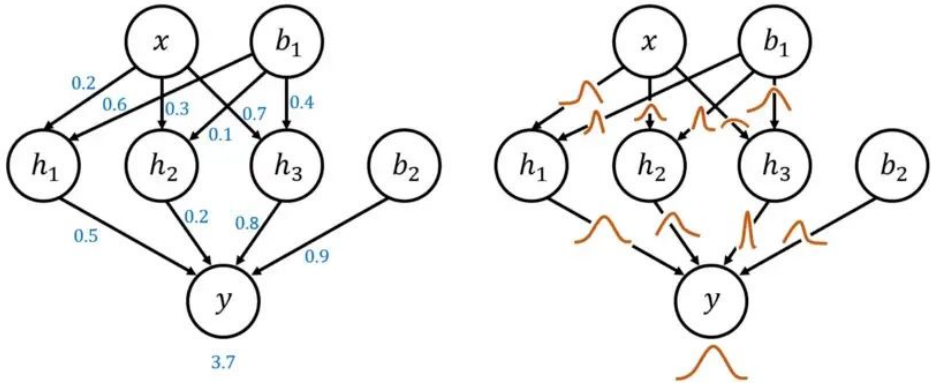


Подходы к обучению бинарных сетей

Обучение бинарных весов параллельно с вещественными

Вероятностные подходы

В рамках вероятностных подходов обучаются стохастические нейронные сети (СНС), в которых для коэффициентов и активаций обучаются не одно конкретное значение, а *распределение* значений.



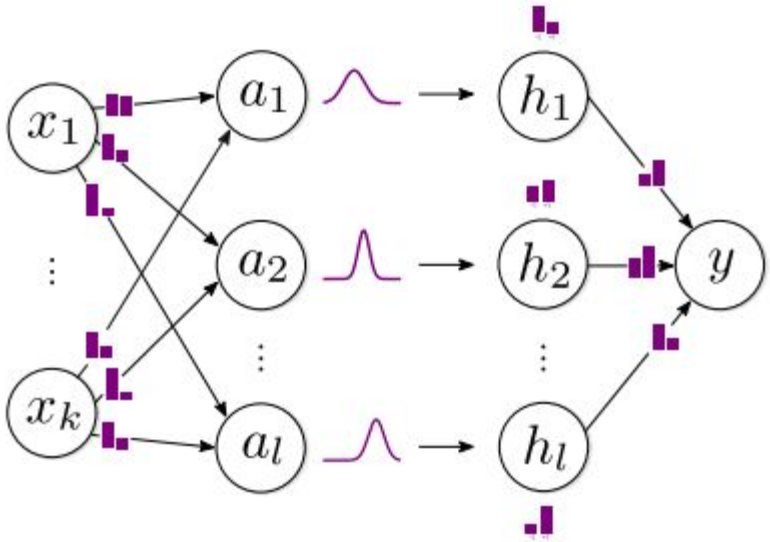
Вероятностные подходы

Каждому весу ставим в соответствие вероятность того, что он равен +1 или -1

Считаем распределение

Аппроксимируем

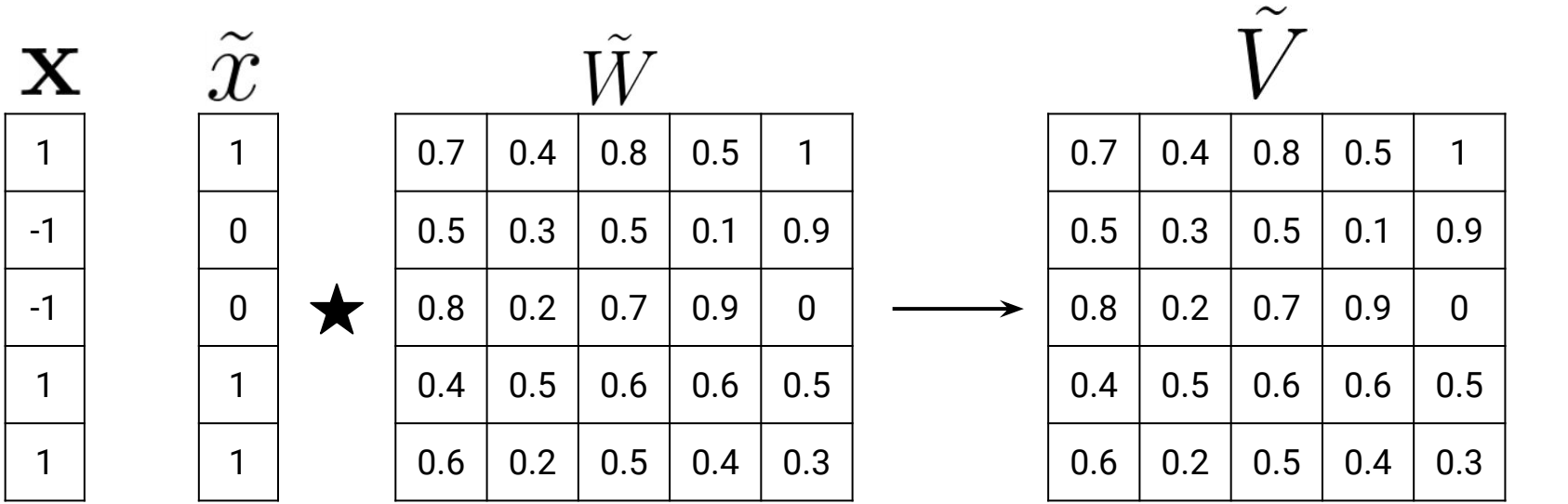
Семплируем



Проблема: использование ЦПТ не всегда обоснованно

Бернуллиевский полносвязный слой

Предлагаемый наивный бернуллиевский слой

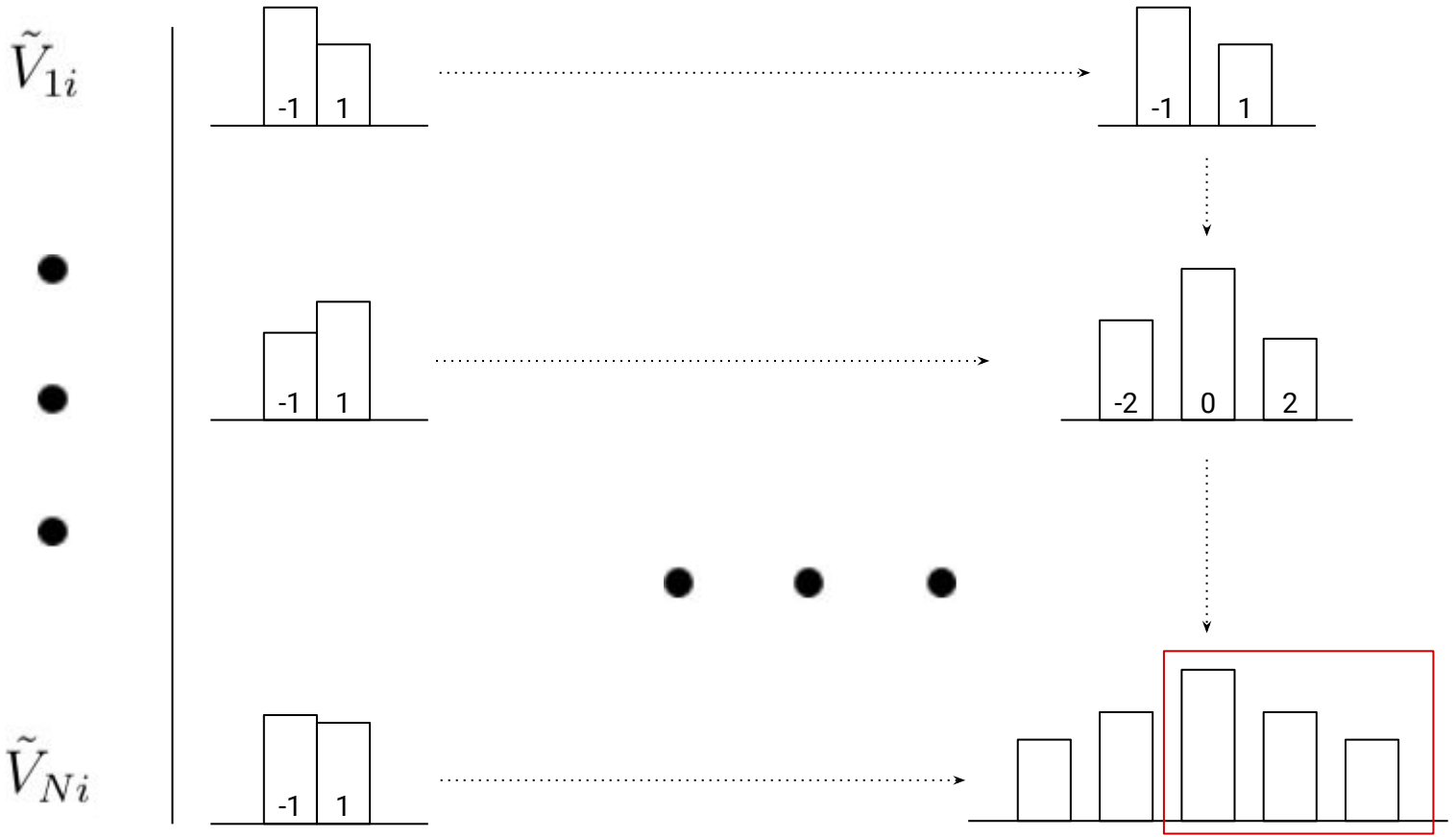


$\tilde{x}_i = P(x_i == 1)$

Weight probabilities matrix

$\tilde{V}_{ij} = \tilde{X}_i \cdot \tilde{W}_{ji} + (1 - \tilde{X}_i) \cdot (1 - \tilde{W}_{ji}).$

Вычисление распределения выхода слоя



Обратное распространение ошибки

Вычисление распределения на последнем шаге:

$$\begin{cases} px_1 = y_1 & p = \tilde{v}_{ijk} \\ px_i + qx_{i-1} = y_i & \\ qx_{N-1} = y_N & q = 1 - p \end{cases}$$

$$2 \le i \le N - 1$$

$$z = \sum_{i=1}^{N/2} y_i$$

$$\frac{\partial L}{\partial \tilde{V}_{ij}} = \frac{\partial L}{\partial \tilde{Z}_{ij}} \cdot \frac{\partial \tilde{Z}_{ij}}{\partial \tilde{V}_{ij}}.$$

$$\Rightarrow \quad \frac{\partial L}{\partial \tilde{V}_{iN}} = g_i \cdot \left(\sum_{k=1}^{N/2} x_{i,k} - \sum_{k-1}^{N/2} x_{i,k-1} \right) = g_i \cdot x_{i,N/2}$$

$$A = \begin{pmatrix} p & 0 & 0 & \dots & 0 \\ q & p & 0 & \dots & 0 \\ 0 & q & p & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 & q \end{pmatrix}$$

$$\Downarrow$$

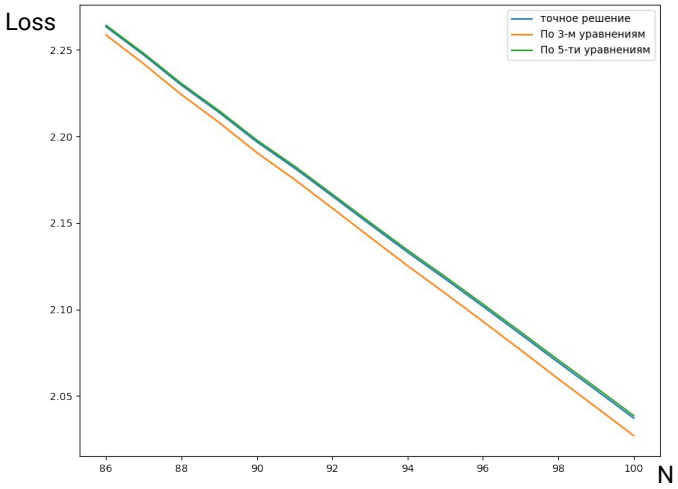
$$\begin{pmatrix} p^2+q^2 & pq & 0 & \dots & 0 \\ pq & p^2+q^2 & pq & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & pq & p^2+q^2 \end{pmatrix}$$

Аппроксимации градиента

Рассмотрим 4 способа вычисления градиента на примере модели из одного полносвязного слоя В качестве градиента для обратного распространения ошибки в этих сетях использованы:

- Точное вычисление градиента через решение системы
- Аппроксимация градиента по трем центральным уравнениям системы
- Аппроксимация градиента по пяти центральным уравнениям
- Модификация вычислений для использования автоградиента

В качестве задачи для моделей сгенерирована случайная матрица весов, которая должна была быть предсказана сетями.
В качестве входа сетям подавались случайные вектора, а в качестве выхода – результаты умножения этих векторов на сгенерированную матрицу.

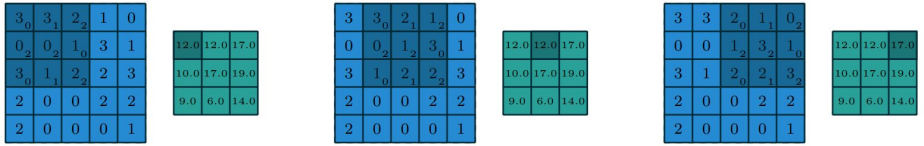


Название	N итераций для схождения
Precise	57.7
Approx1	63.4
Approx2	59.2
Autograd	57.3

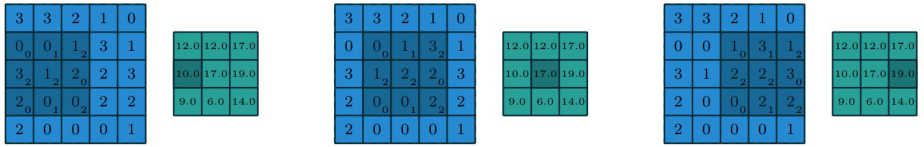
Построение модели

Сверточный слой

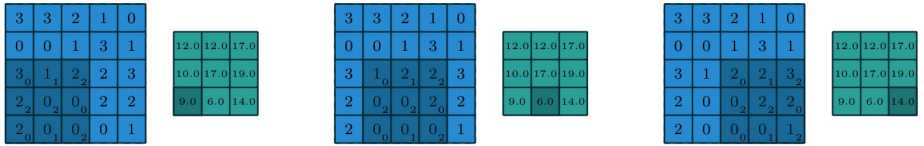
Операция свертки:



Операция свертки может быть представлена как операция матричного умножения.



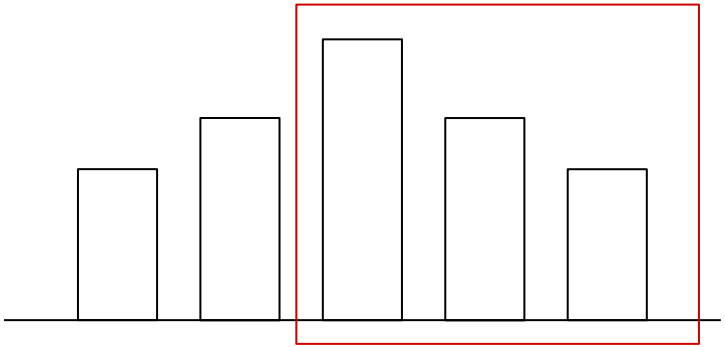
Таким образом, мы можем применить подход для полносвязных слоев.



$$\text{out}(N_i, C_{\text{out}_j}) = \text{bias}(C_{\text{out}_j}) + \sum_{k=0}^{C_{\text{in}}-1} \text{weight}(C_{\text{out}_j}, k) \star \text{input}(N_i, k)$$

Слой матожидания

В полносвязном слое на предпоследнем шаге мы получаем следующее распределение элемента выходного вектора

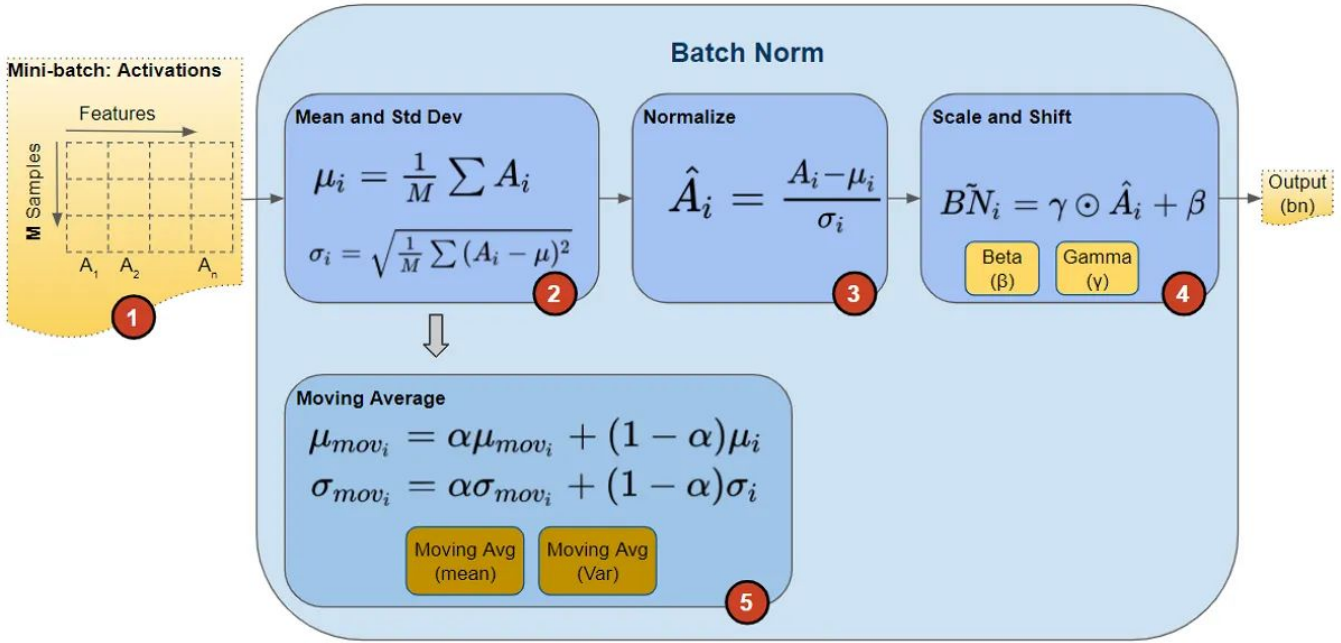


Чтобы получить вероятность для бинарного вектора мы должны просуммировать вероятности получить неотрицательное значение

Однако на последнем слое мы хотим получить вещественное предсказание класса. Поэтому мы считаем математическое ожидание выхода.

Batch normalization

Батчнорм делает обучение более быстрым и стабильным за счет нормализации входных данных путем повторного центрирования и масштабирования.
Однако после нормализации мы теряем первоначальное значение вывода слоя – вероятность. Поэтому нам нужно было вернуть наши выходные данные обратно в распределения вероятностей. Эта проблема была решена путем добавления сигмоиды в качестве функции активации



Бинаризация

Подход к бинаризации

Регуляризация
Позволяет нам приблизить вероятности к 0 и 1.
Таким образом модель будет “более уверенной” в своих весах.

Частичная бинаризация весов с дообучением

Функция активации
Сигмоида должна стать знаком в бинарной модели.
Для этого мы ввели и увеличивали параметр температуры: **sigmoid(x / (T + epsilon))**

Архитектура сетей

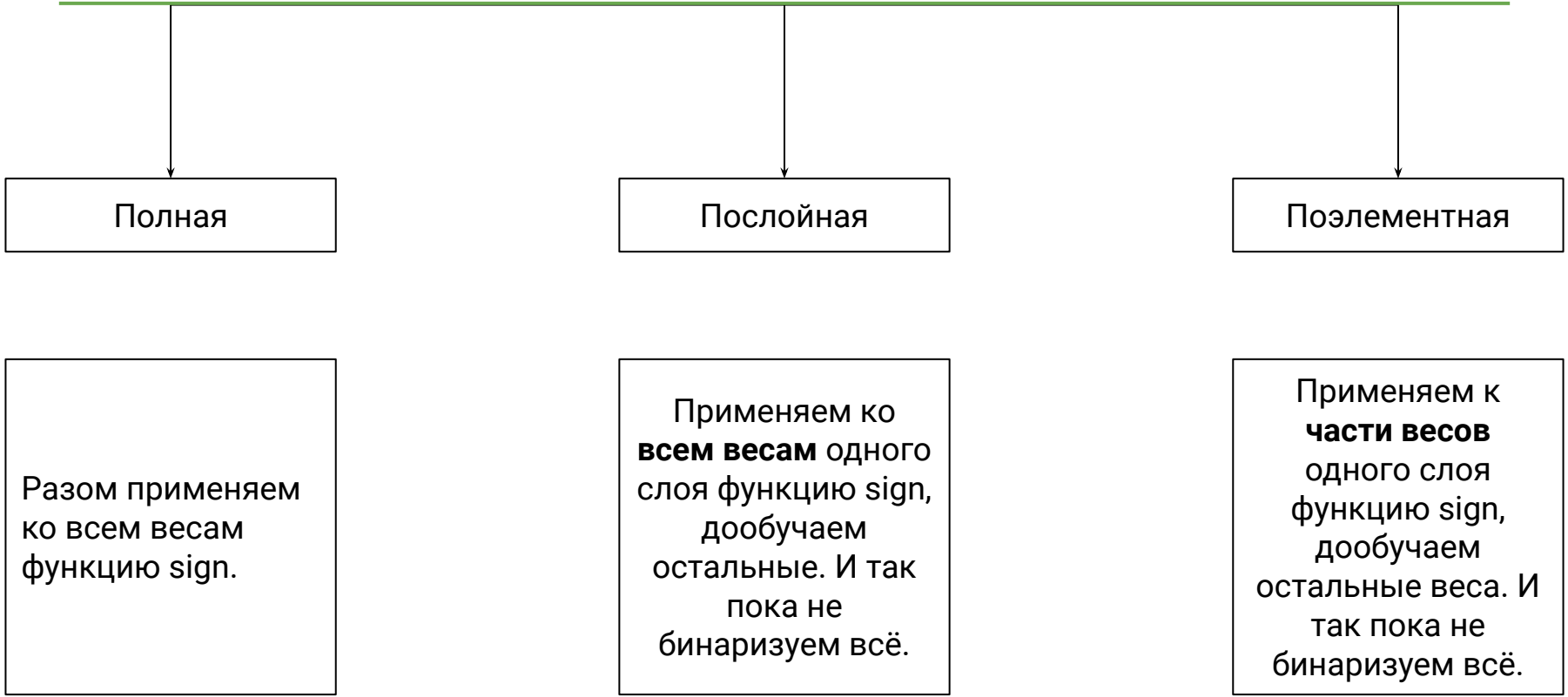
Наша модель

Convolutional(in=1, out=8, kernel=6x6, stride=2**)**
BatchNorm2d(features=8**)**
Sigmoid(x / Temp + 0.0001**)**
Convolutional(in=8, out=32, kernel=10x10, stride=2**)**
BatchNorm2d(features=32**)**
Sigmoid(x / Temp + 0.0001**)**
FullyConnected(in=32, out=64)
BatchNorm2d(features=64**)**
Sigmoid(x / Temp + 0.0001**)**
FullyConnected(in=64, out=80**)**
BatchNorm1d(features=80**)**
Sigmoid(x / Temp + 0.0001**)**
MathExpectationLayer(in=80, out=10**)**

STE

Convolutional(in=1, out=8, kernel=6x6, stride=2**)**
BatchNorm2d(features=8**)**
Sign(x)
Convolutional(in=8, out=32, kernel=10x10, stride=2**)**
BatchNorm2d(features=32**)**
Sign(x)
FullyConnected(in=32, out=64)
BatchNorm2d(features=64**)**
Sign(x)
FullyConnected(in=64, out=80**)**
BatchNorm1d(features=80**)**
Sign(x)
MathExpectationLayer(in=80, out=10**)**

Бинаризация наивной бернуллиевской сети



Эксперименты

MNIST

Это датасет рукописных цифр, который мы предварительно бинаризовали.

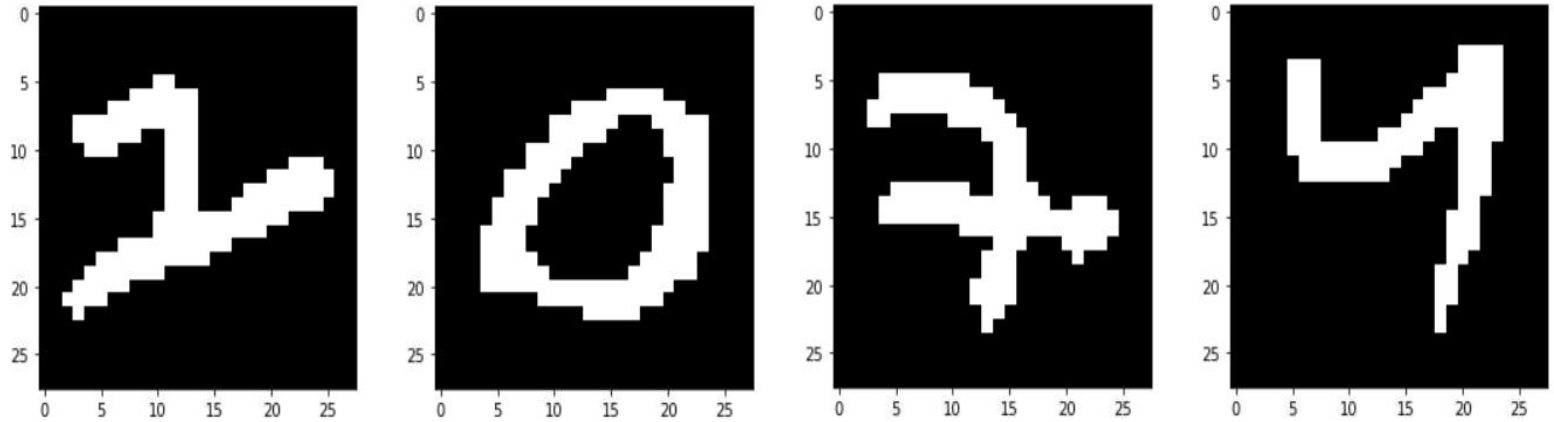
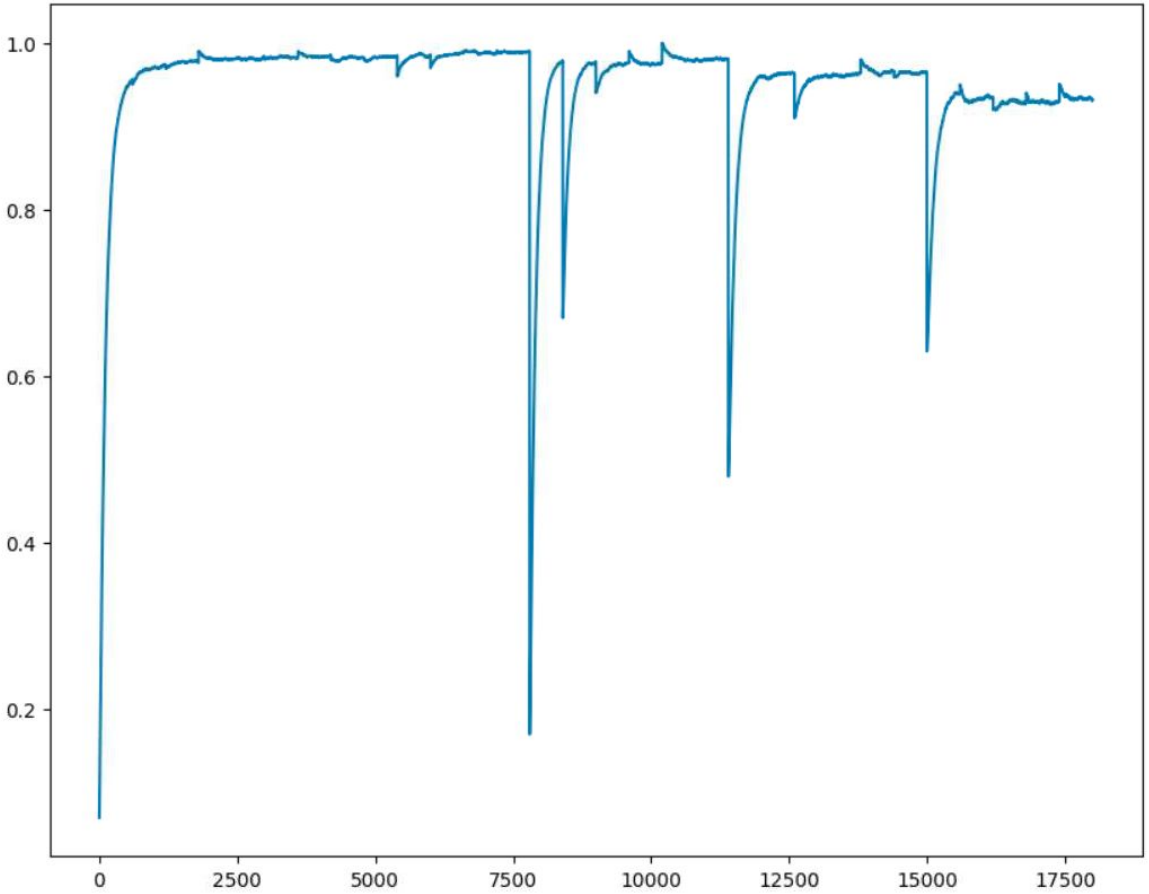
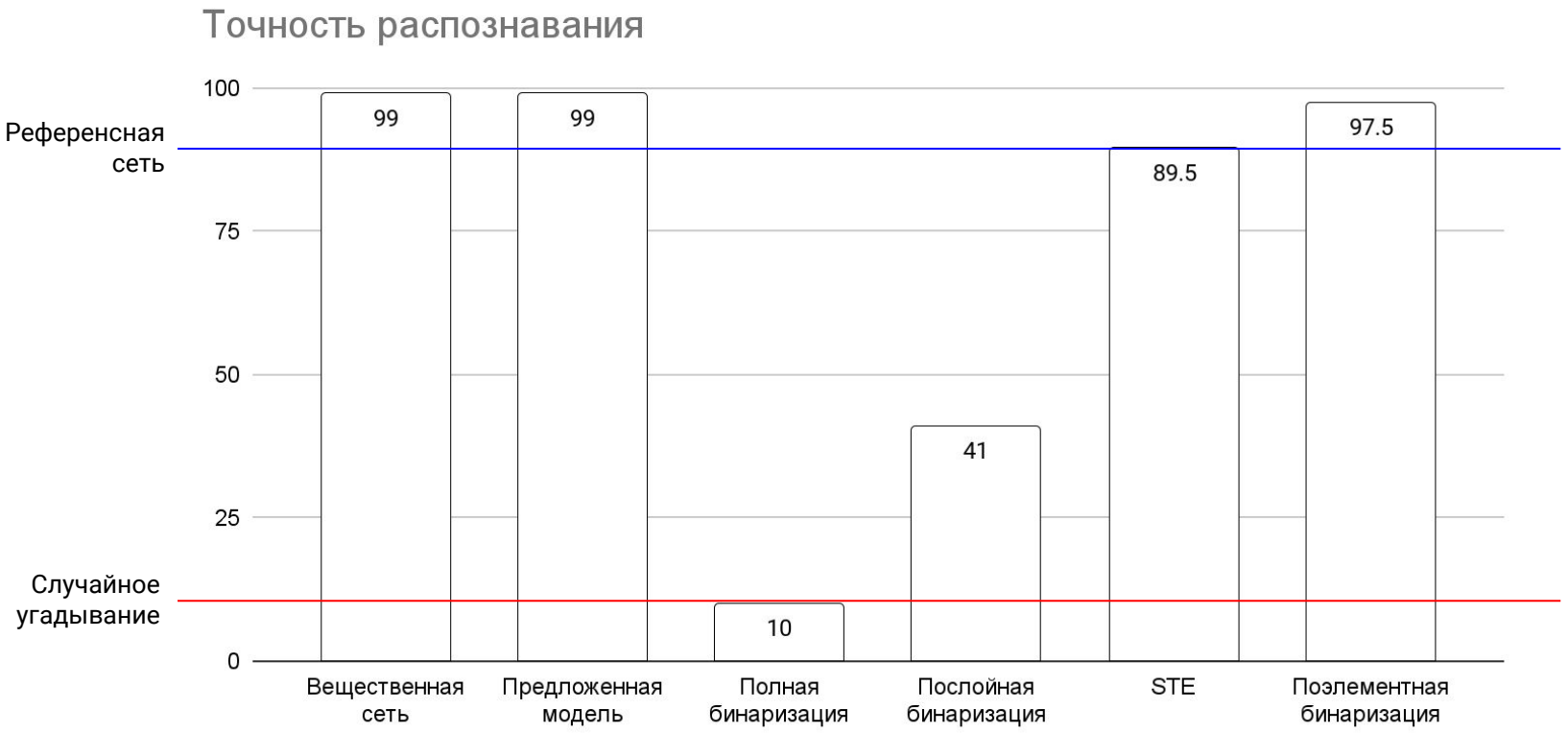


График обучения и бинаризации модели



Результаты экспериментов



Заключение

Предложены в работе:

- наивная бернуллиевская аппроксимация вещественной сети;
- метод бинаризации для этой аппроксимации;
- различные аппроксимации градиента. Для модели выбрана наилучшая по параметрам точность/скорость: аппроксимация по трем центральным уравнениям;

Получены численные результаты на наборе данных MNIST:

- 99% точности у классической вещественной сети;
- 97.5% точности предложенной бинарной модели;
- 89.5% точности у референсной бинарной модели, полученной с помощью STE;

Результаты представлены на 65-й Конференции МФТИ. Итог – диплом 3 степени.

Бинарные нейронные сети

Один из способов решения этих проблем: использование бинарных нейронных сетей – нейронных сетей с бинарными весами и активациями.

Плюсы
Позволяют быструю реализацию на основе битовых операций
Требуют в ~32 раза меньше памяти для хранения коэффициентов

Минусы
Уступают по качеству вещественным моделям: 1. Методы обучения как правило разрабатываются эмпирически 2. Требуют подбора способа обучения в каждой конкретной задаче

Можно использовать сложные сети на маломощных устройствах

Разработка эффективных методов обучения бинарных сетей представляет значительный научный и практический интерес