МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

ОРДЕНА ТРУДОВОГО КРАСНОГО ЗНАМЕНИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ»

(МТУСИ)

ОТЧЕТ

по курсовой работе

по дисциплине: «Системы искусственного интеллекта»

Тема: «Разработка авторегрессионной генеративной модели (VAR) на основе собственной реализации нейросетевого фреймворка»

Выполнил студент: Быков Д.В

Группа: БВТ2201

Москва, 2025

Содержание

1 Введение

В последние годы глубокое обучение достигло значительных успехов не только в задачах распознавания образов (дискриминативные модели), но и в задачах создания новых данных. Этот класс алгоритмов называется **генеративными моделями**.

В отличие от дискриминативных моделей, которые обучаются предсказывать целевую переменную y по входным данным x (то есть моделируют условную вероятность $P(y|x)$), генеративные модели стремятся выучить распределение самих данных $P(x)$. Это позволяет им генерировать новые образцы, похожие на те, что были в обучающей выборке, но не повторяющие их точь-в-точь.

Существует несколько основных типов генеративных моделей:

1. **GAN (Generative Adversarial Networks):** соревнование генератора и дискриминатора.
2. **VAE (Variational Autoencoders):** вероятностное сжатие данных с регуляризацией скрытого пространства.
3. **Диффузионные модели (DDPM):** постепенное удаление шума из сигнала.
4. **Авторегрессионные модели (VAR/AR):** моделирование данных как последовательности, где каждый элемент зависит от предыдущих.

В данной курсовой работе рассматривается последний тип — **авторегрессионные модели (VAR)**. Основная задача работы — не просто использование готовых библиотек, а реализация низкоуровневых компонентов нейронной сети (слоев, функций активации и алгоритма обратного распространения ошибки) «с нуля» для построения архитектуры типа MADE (Masked Autoencoder for Distribution Estimation).

2 1. Теоретическая часть

2.1 1.1. Принцип авторегрессионного моделирования

Авторегрессионные модели основаны на цепном правиле вероятности. Если мы рассматриваем изображение x как вектор из n пикселей $x = (x_1, x_2, ..., x_n)$, то совместную вероятность появления такого изображения можно разложить на произведение условных вероятностей:

$$P(x) = \prod_{i=1}^{n} P(x_i \mid x_1, x_2, ..., x_{i-1})$$

Это означает, что значение i-го пикселя зависит только от всех предыдущих пикселей $x_{\{<i\}}$. При генерации мы создаем пиксели последовательно: сначала x_1, затем x_2 на основе x_1, и так далее.

2.2 1.2. Проблема стандартных автоэнкодеров

Классический автоэнкодер (Autoencoder) принимает на вход весь вектор x и учится восстанавливать его же. В стандартном полносвязном слое каждый выходной нейрон связан со всеми входными нейронами. Если использовать такую архитектуру для генерации, сеть просто «скопирует» вход на выход, подсматривая значения «будущих» пикселей, что нарушает принцип причинности (causality).

2.3 1.3. Решение: MADE (Masked Autoencoder)

Для соблюдения свойства авторегрессии в архитектуре многослойного перцептрона (MLP) применяется метод маскирования весов. Идея заключается в изменении матрицы весов W слоя путем поэлементного умножения на бинарную маску M:

$$W_{\text{masked}} = W \odot M$$

Где $M_{\{j,k\}} = 1$, если k-й вход разрешен для вычисления j-го выхода, и 0 в противном случае. В данной работе реализуется строгая последовательная маска:

- Для входного слоя (Тип A): нейрон i на выходе слоя не может видеть нейрон i на входе (чтобы не подсматривать свой же цвет).

- Для скрытых слоев (Тип B): нейрон i может видеть нейроны $j \leq i$.

3 Практическая часть

3.1 Реализация базовых компонентов

Ключевой особенностью данной работы является отказ от использования автоматического дифференцирования (`autograd`) для реализации слоев. Был разработан собственный мини-фреймворк, включающий классы `Layer`, `Linear`, `ReLU`, `Sigmoid`, `BCELoss` и оптимизатор `SGD`.

Пример реализации метода `backward` для полносвязного слоя, который вычисляет градиенты согласно цепному правилу (Chain Rule):

```
def backward(self, grad_output):
    # grad_output shape: (Batch, Out)

    # dL/dW = X^T * dL/dY
    self.grads['W'] = self.x.T @ grad_output

    # dL/db = sum(dL/dY)
    self.grads['b'] = torch.sum(grad_output, dim=0)

    # dL/dX = dL/dY * W^T
    grad_input = grad_output @ self.params['W'].T

    return grad_input
```

Такой подход позволяет глубже понять математику обучения нейронных сетей и проконтролировать процесс обновления весов.

3.2 Реализация слоя MaskedLinear

На основе базового класса Linear был создан слой MaskedLinear. В методе `__init__` создается нижнетреугольная матрица-маска.В методе forward перед умножением матриц происходит наложение маски:

```
self.params['W'].data = self.params['W'] `*` self.mask
return super().forward(x)
```

В методе backward градиенты весов также умножаются на маску, чтобы предотвратить изменение запрещенных связей.

3.3 Архитектура и обучение модели

Модель представляет собой MLP следующей структуры:

- Вход (784 пикселя, MNIST).
- MaskedLinear (Type A) -> ReLU.
- MaskedLinear (Type B) -> ReLU.
- MaskedLinear (Type B) -> Sigmoid.

Использовался датасет MNIST, предварительно бинаризованный (значения пикселей приведены к 0 или 1). Функция потерь — бинарная кросс-энтропия (BCE Loss). Обучение проводилось с помощью собственного оптимизатора SGD на GPU.

4 Результаты работы

4.1 Динамика обучения

В процессе обучения наблюдалось стабильное снижение функции потерь. График обучения демонстрирует, что реализованный вручную алгоритм обратного распространения ошибки работает корректно.

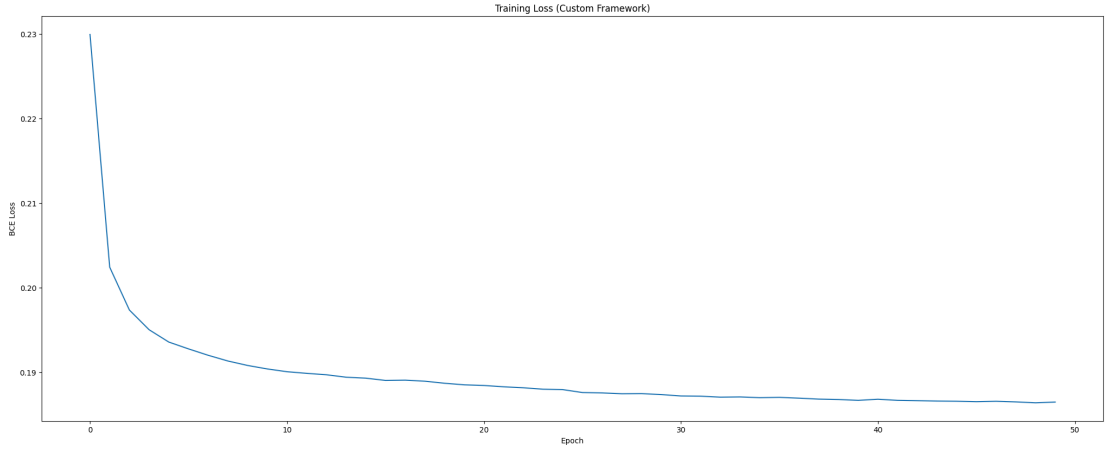

Рис. 1. График функции потерь по эпохам

4.2 Генерация изображений

Для проверки генеративной способности использовался метод попиксельного сэмплирования. Генерация начинается с пустого изображения. На

каждом шаге i (от 0 до 783) модель предсказывает вероятность того, что i-й пиксель будет белым, основываясь на уже сгенерированных пикселях $0...i-1$.

Результаты генерации представлены ниже:

Рис. 2. Примеры цифр, сгенерированных VAR моделью

Полученные изображения имеют четкую структуру и напоминают рукописные цифры из датасета MNIST, что подтверждает корректную работу авторегрессионного механизма.

5 Заключение

В ходе выполнения курсовой работы была успешно реализована и исследована авторегрессионная генеративная модель (VAR) архитектуры MADE. Основные достижения: Разработан собственный инструментарий для глубокого обучения: реализованы слои Linear, функции активации ReLU, Sigmoid и механизм ручного расчета градиентов (Backpropagation). Реализован слой MaskedLinear, обеспечивающий соблюдение принципа причинности в полносвязной сети. Обучена модель на датасете MNIST, способная генерировать новые валидные изображения цифр.