Технология дистилляции знаний для обучения нейронных сетей на примере задачи бинарной классификации

Бирюкова В.А.[1]

С использованием технологии обучения нейронных сетей, дистилляции знаний, были получены модели, решающие задачу бинарной классификации с производительностью, превышающей производительность сети-учителя примерно в 5 раз при несущественном падении качества. Сверточная нейронная сеть ResNet-18 была обучена двумя способами по данной технологии (с помощью предобученной сети ResNet-50) и классическим методом. Введено понятие степени неуверенности модели на множестве объектов как величины отклонения предсказаний нейронной сети от принимаемых за ответ значений. Были также проведены эксперименты по рекурсивному применению технологии дистилляции знаний.

Ключевые слова: дистилляция знаний, бинарная классификация, остаточная нейронная сеть, сверточная нейронная сеть, степень неуверенности модели на множестве объектов, рекурсивное обучение нейронных сетей.

1. Введение

В настоящее время благодаря появлению технологии сверточных нейронных сетей и высокопроизводительных средств бурно развивается направление построения искусственных систем для распознавания визуальных образов [1],[2],[3],[4]. При этом трудоемким является процесс подготовки обучающей базы, как правило, требующей большого количества вручную размеченных изображений, что сильно увеличивает стоимость

[1] *Бирюкова Вероника Андреевна* — студент каф. высшей математики института кибернетики РТУ МИРЭА, e-mail: biryukovaveronika@mail.ru.

Biryukova Veronika Andreevna — student, MIREA - Russian Technological University, Institute of Cybernetics, Chair of Higher Mathematics.

разработок. Также возникает необходимость размещения на портативных устройствах небольших нейронных сетей, способных выдавать достаточно точные результаты [5]. Поэтому для решения ряда задач может быть использована технология обучения относительно простых сетей на базе изображений без ручной разметки, но с использованием предварительно обученной более мощной модели. Для этого разработана методика обучения нейронных сетей – дистилляция знаний (Knowledge Distillation) [6],[7]: пусть имеется мощная модель (или ансамбль моделей), которая обучена улавливать сложные признаки (features) и выдавать достаточно точные предсказания. Такую нейронную сеть называют сетью-учителем. С её помощью обучается более простая модель (сеть-студент), которую можно обучить так, что она будет давать результаты, сопоставимые с результатами сети-учителя.

Целью данной работы является исследование применения технологии Knowledge Distillation в рамках задачи бинарной классификации. В этой работе сверточная нейронная сеть ResNet-18 («остаточная» нейронная сеть [8]) была обучена по трем сценариям. Сеть, обученную по первому сценарию, то есть она обучалась на базе данных с ручной разметкой, в дальнейшем будем называть классическим студентом. Второй и третий сценарии предполагают обучение по меткам, которые являются предсказаниями сети, полученной из предобученной сети ResNet-50 [8], причем в разных сценариях эти метки использовались в различных формах представления. Далее эти модели будем называть soft-студент и hard-студент. ResNet-50 была взята из нейросетевой библиотеки Keras. Полученные результаты были изучены, а также были проведены эксперименты с рекурсивным применением исследуемой технологии. Эксперименты проводились на базе изображений кошек и собак Kaggle «Dogs vs. Cats» [9].

2. Основные понятия

Машинное обучение (англ. machine learning, ML) — класс методов искусственного интеллекта, целью которых является создание математических моделей, способных выполнять конкретные задачи без использования явных инструкций, а основываясь на выявленных в процессе обучения эмпирических закономерностях в данных, на которых она обучалась.

Искусственная нейронная сеть (нейронная сеть) или нейросетевая модель – это математическая модель, построенная по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма. У модели данного типа есть тренируемые параметры, которые меняются во время процедуры обучения, и гиперпараметры, параметры модели, требующие ручной настройки и задаваемые до процесса обучения. Тренируемые параметры, далее именуемые параметрами, – это веса и смещения (weights and biases) [10]. Гиперпараметрами являются, например, архитектура сети (строение модели), функция потерь, оптимизатор. Общие понятия, связанные с нейронными сетями, можно посмотреть в работе [10]. В этом исследовании рассматриваются сверточные нейронные сети специального вида, называемые «остаточными» нейронными сетями, ResNet [8]. Эти сети используют дополнительные соединения между слоями для нивелирования эффекта «затухающего градиента» [11]. С помощью нейронных сетей можно решать различные задачи машинного обучения, например, задачу классификации.

Постановка задачи классификации. Дана обучающая база – множество объектов (изображений, аудиофайлов и т.д.). Каждому элементу этого множества соответствует некоторый класс (возможны варианты задачи, где один объект сопоставляется нескольким классам). Требуется, чтобы модель после процесса обучения на этой базе могла правильно предсказывать принадлежность объектов, которые не использовались во время обучения, к одному из заданных классов.

Если работа ведется с изображениями, то эта задача будет относиться к области компьютерного зрения «Computer Vision» [12]. Компьютерное зрение – это теория и соответствующая реализация интеллектуальных систем [13], способных извлекать полезную информацию из изображений.

Одним из способов обучения моделей и, в частности нейронных сетей, является метод «обучения с учителем» [10]. Для обучения модели используются множество объектов и множество меток. Обучающим множеством называется совокупность пар «объект, метка», между составляющими которых существует некоторая зависимость. На основе этих данных требуется восстановить эту зависимость, то есть построить модель, способную для любого возможного корректного входного изображения выдать достаточно точный ответ (предсказание). Важно, чтобы обучаемая модель была способна к обобщению, то есть к адекватному отклику на данные, выходящие за пределы обучающего множества. Если же мо-

дель научилась достаточно точно отвечать на объекты множества, на котором она обучалась, а на новые изображения – нет, то такой эффект называется «переобучением» модели [10].

Для обучения модели обучающее множество делится на тренировочное, проверочное и тестовое множества. В ходе обучения нейронной сети объекты из тренировочного множества поступают на вход модели, а на выходе получаются соответствующие значения, которые сравниваются с метками, выступающими в качестве «правильных» ответов, посредством чего вычисляется величина ошибки нейронной сети с помощью заранее выбранной функции потерь (Loss function). После этого осуществляется процесс обратного распространения ошибки [10], благодаря чему получают значения для корректировки параметров сети. Как именно корректируются параметры, зависит от важного механизма – оптимизатора. После корректировки параметров процедура обучения повторяется, пока не выполнится заданный критерий остановки.

Как правило, обучение проходит по «эпохам». Эпохой называется часть обучающего процесса, во время которого используется всё тренировочное множество. По итогам эпохи обычно оценивают качество работы сети на проверочном множестве [14]. После этого начинается новая эпоха, и тренировочное множество изображений снова обрабатывается моделью. Как правило, в течение одной эпохи тренировочные объекты подаются не по одному и не все сразу, а порциями. То есть на вход сети поступает какая-то выборка (mini-batch) из тренировочного множества, она целиком проходит через сеть, и только после этого начинается процедура корректировки параметров. Размер выборки определяется заранее, таким образом это гиперпараметр сети.

Обычно для улучшения результатов обучения, перед тем как поступить на вход модели, изображения проходят процедуру предобработки, например: значения пикселей нормируются, или картинки приводятся к соответствующему разрешению. Эти преобразования применяются к каждому изображению. Также часто в процессе обучения модели используется аугментация данных – искусственное увеличение обучающего множества за счет преобразования имеющихся изображений, при котором полученное изображение продолжает принадлежать классу исходного объекта [15]. В этом случае при каждом вызове изображение меняется случайным образом в рамках заданных преобразований. Аугментация данных позволяет минимизировать возможность переобучения и способствует обучению нейронной сети обобщать образы.

За результат обучения принимается нейронная сеть с теми значениями параметров, при которых модель показала наилучшее качество работы на проверочном множестве.

После процесса обучения оценивается качество работы сети на тестовом множестве, то есть на тех изображениях, которые модель не обрабатывала в процессе обучения. Тем самым определяется итоговая оценка результативности модели.

Для задачи бинарной классификации в данной работе используется бинарная кросс-энтропия (Binary Cross-Entropy) [16]. Эта функция выводится с помощью метода максимального правдоподобия. Формула данного выражения для одного объекта:

$$CE(p, y) = -(y \cdot \log(p) + (1 - y) \cdot \log(1 - p)), \tag{1}$$

где y и p – метка и предсказание сети на этом объекте соответственно.

Для выборки из n объектов:

$$Loss(\bar{p}, \bar{y}) = \frac{1}{n} \sum_{i=1}^{n} CE(p_i, y_i), \tag{2}$$

где $\bar{p}$ – вектор предсказаний нейронной сети на n объектах с компонентами p_i (предсказание модели на i-ом изображении); $\bar{y}$ – вектор меток этих n объектов с компонентами y_i (метка i-ого изображения).

В данном исследовании в качестве оптимизатора применяется алгоритм Adam (adaptive moment estimation) [17], сочетающий в себе следующие две идеи. Во-первых, идею из численных методов о накоплении градиента, «если некоторое время двигаться в определённом направлении, то следует туда двигаться некоторое время и в будущем». Во-вторых, идею более слабого обновления параметров как реакции на типичные признаки для какого-нибудь из заданных классов, выявленные на объекте из обрабатываемой выборки.

Параметры сети корректируются на каждой t-ой выборке, номер этой выборки меняется в следующих пределах:

$$t \in \{1, \ldots, P \cdot S\}, \tag{3}$$

где P – количество эпох, S – количество выборок, обрабатываемых сетью за одну эпоху.

Этот оптимизатор использует понятия экспоненциально взвешенного скользящего среднего для градиента функции потерь на t-ой выборке m_t:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \ , \tag{4}$$

где g_t – градиент функции потерь, β_1 – гиперпараметр сети, обычно полагается $\beta_1 = 0.9$;

и оценки средней нецентрированной дисперсии на t-ой выборке υ_t:

$$\upsilon_t = \beta_2 \upsilon_{t-1} + (1 - \beta_2) g_t^2 \ , \tag{5}$$

где g_t – градиент функции потерь, β_2 – гиперпараметр сети, обычно полагается $\beta_2 = 0.999$.

Как правило, напрямую оценку дисперсии и скользящее среднее не используют. Их искусственно увеличивают на первых выборках. Общие формулы для рассматриваемых величин следующие:

$$\widehat{m_t} = \begin{cases} \frac{m_t}{1-\beta_1^t}, & \text{если } 0 < t < k_1 \\ m_t \quad , & \text{если } t \geq k_1 \end{cases} \ , \tag{6}$$

$$\widehat{\upsilon_t} = \begin{cases} \frac{\upsilon_t}{1-\beta_2^t}, & \text{если } 0 < t < k_2 \\ \upsilon_t \quad , & \text{если } t \geq k_2 \end{cases} \ , \tag{7}$$

где k_1 и k_2 часто полагаются 10 и 1000 соответственно.

Скорость обучения – это параметр обучения нейронных сетей, позволяющий управлять величиной коррекции весов на каждой выборке.

В алгоритме Adam этот параметр меняется по следующей формуле:

$$\eta_t = \eta_0 \cdot \frac{\sqrt{1 - \beta_2^t}}{1 - \beta_1^t} \ , \tag{8}$$

где η_0 – начальное значение скорости обучения.

В библиотеке Keras реализован механизм дополнительного изменения скорости обучения в этом оптимизаторе с помощью задаваемого до обучения параметра d:

$$\eta_t = \eta_0 \cdot \frac{1}{1 + d \cdot (t-1)} \cdot \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t} \ . \tag{9}$$

Формула обновления параметра сети на t-ой выборке θ_t:

$$\theta_t = \theta_{t-1} - \frac{\eta_t}{\sqrt{\widehat{v_t}} + \epsilon} \widehat{m_t} \ , \tag{10}$$

где θ_{t-1} – значение этого параметра после обновления на $(t-1)$-ой выборке, ϵ – сглаживающий параметр (гиперпараметр сети), который в данной работе полагается равным $\epsilon = 10^{-7}$. Этот гиперпараметр необходим, чтобы избежать деления на 0.

Для оценки качества работы нейронной сети используется оценка точности (accuracy), которую максимизируют в процессе обучения. Для задачи бинарной классификации на одном объекте эта оценка вычисляется по следующей формуле:

$$acc(p, y) = \begin{cases} 1, \text{ если } (p > th \text{ и } y > th) \text{ или } (p \leq th \text{ и } y \leq th) \\ 0, \text{ если } (p > th \text{ и } y \leq th) \text{ или } (p \leq th \text{ и } y > th) \end{cases} \ , \tag{11}$$

где th – пороговое значение, которое в данной работе равно 0.5; метка объекта y и предсказание модели на этом объекте p являются действительными числами из отрезка $[0, 1]$.

Для выборки из n объектов (оценивается следующим образом):

$$acc(\bar{p}, \bar{y}) = \frac{1}{n} \sum_{i=1}^{n} acc(p_i, y_i) \ , \tag{12}$$

где $\bar{p}$ – вектор предсказаний нейронной сети на n объектах, с компонентами p_i (предсказание модели на i-ом изображении); $\bar{y}$ – вектор меток этих n объектов, с компонентами y_i (метка i-ого изображения) .

3. Обучение нейронных сетей

База данных, использованная для задачи бинарной классификации, представляет собой 25 000 изображений кошек и собак (12 500 для каждого класса) с сайта Kaggle [9]. Во всех экспериментах полагается, что, если сеть выдает значение строго больше 0.5, то она классифицирует этот объект как «изображение с собакой». В противном случае, если выходное значение модели меньше или равно 0.5, то нейронная сеть предсказывает принадлежность данного изображения к классу «изображение с кошкой». Всё множество изображений было поделено на тренировочное (70%), проверочное (15%) и тестовое (15%) множества. Таким образом, количество тренировочных объектов составило 17500 изображений. Проверочных – 3750. Тестовых – 3750. При работе со всеми моделями изображения загружались в формате «RGB» в виде четырехмерного массива (размер выборки × высота × ширина × количество цветовых каналов). Поскольку использовался формат «RGB», то количество цветовых каналов равнялось трем. В каждой ячейке этого массива указывалось значение одного из цветовых каналов соответствующего пикселя изображения, равное целому числу из диапазона $[0, 255]$. Для каждой модели использовалась одинаковая предобработка данных:

1) Исходное изображение преобразовывалось к разрешению 224×224 пикселей с помощью интерполяционного метода ближайшего соседа.

2) Производилась нормировка значений массива, то есть каждый его элемент был поделен на 255, после всех преобразований предобработки и аугментации.

При обучении hard- и soft-студентов в качестве меток i-ого изображения y_i выступили значения, сформированные по значению предсказания сети-учителя на этом объекте p_i. Для hard-студента:

$$y_i = \begin{cases} 1, & \text{если } p_i > 0.5 \\ 0, & \text{если } p_i \leq 0.5 \end{cases}, \tag{13}$$

а для soft-студента:

$$y_i = p_i\,. \tag{14}$$

Все модели оценивались на тестовом множестве с ручной разметкой с помощью двух генераторов изображений: один генератор подавал на вход нейронной сети картинки, которые прошли только процедуру предобработки, а другой генератор для оценки качества работы нейронной сети изменял картинки, как и генераторы, использовавшиеся при обучении этой модели, то есть с применением преобразований для аугментации данных. Для сетей, которые обучались не на базе с ручной разметкой, проводилась также оценка на тестовом множестве с разметкой сети-учителя – как с аугментацией, так и без неё.

Генераторы, использовавшие помимо предобработки преобразования для аугментации данных, меняли исходные изображения случайным образом, поэтому оценивание проводилось несколько раз. В таблицах с результатами представлены наилучшие значения для функции потерь согласно формуле (2) и для оценки качества работы сети в соответствии с формулой (12).

3.1. Обучение сети-учителя

В роли сети-учителя выступала нейронная сеть ResNet-50, преобразованная в соответствии с поставленной задачей: использовалась только часть сети, отвечающая за выделение признаков на изображении. К ней был добавлен слой Dropout [10] с долей «забывания» равной 0.5 и полносвязный слой [10] с одним нейроном. К выходу этого слоя применили сигмоидальную функцию активации [10]. Схема сети-учителя изображена на рисунке 1:

В качестве оптимизатора сети использовался метод Adam со скоростью обучения 10^{-6} и значением параметра d из формулы (9) равным 0.01. Параметры β_1 и β_2 из формул (4)–(9) равнялись соответственно 0.9 и 0.999. Функция потерь – бинарная кросс-энтропия. Для оценки качества работы нейронной сети вычислялась точность модели в соответствии с формулой (12). При обучении этой нейронной сети для аугментации данных использовалась симметрия относительно вертикальной оси, проходящей через центр изображения. Обучение модели проходило в течение 25 эпох с выборкой размера 32. Динамика значений функции потерь и точности представлена на рисунке 2:

Результаты обучения сети-учителя:

```
_________________________________________________
Layer (type)                 Output Shape
=================================================
input_3 (InputLayer)         (None, 224, 224, 3)
_________________________________________________
resnet50 (Model)             (None, 2048)
_________________________________________________
dropout_3 (Dropout)          (None, 2048)
_________________________________________________
dense_3 (Dense)              (None, 1)
_________________________________________________
activation_3 (Activation)    (None, 1)
=================================================
```

Рис. 1. Схема сети-учителя, полученной из ResNet-50

Множество \ Функция		Функция потерь	Точность
Тренировочное множество (лучшее значение в процессе обучения)		0.00079	0.99983
Проверочное множество (лучшее значение в процессе обучения)		0.09638	0.97526
Тестовое множество с ручной разметкой	Без аугментации	0.10340	0.97333
	С аугментацией	0.10310	0.97547

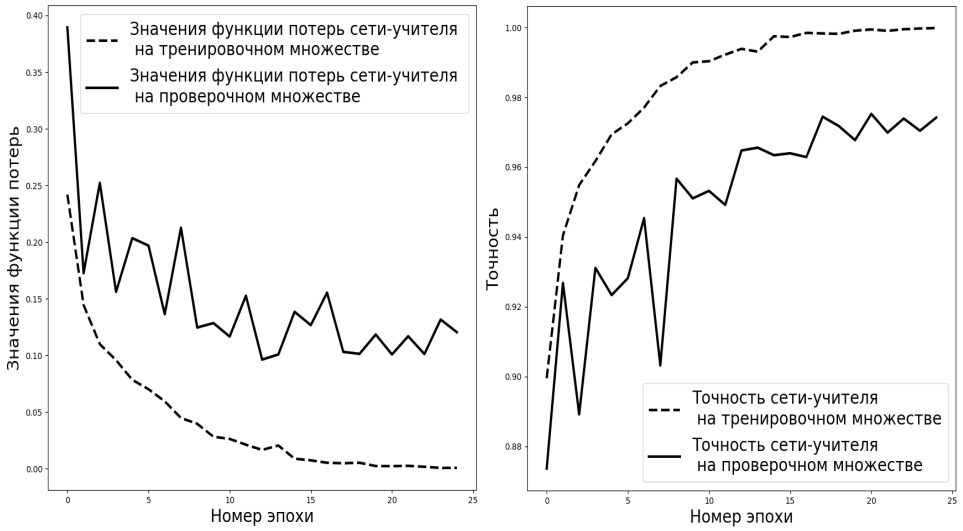

Рис. 2. Динамика значений функции потерь и точности сети-учителя

3.2. Обучение студентов

Модель ResNet-18 была реализована в библиотеке Keras с архитектурой из работы [8]. После слоя Average Pooling [10] его выход (массив 1x1x512) был преобразован к вектору размерности 512. К нему был присоединен полносвязный слой с одним нейроном. К выходу этого слоя применялась сигмоидальная функция активации.

Данная модель была обучена трижды: классическим способом, и два раза при обучении метки формировались по предсказаниям сети-учителя. В первом случае использовалась формула (13), а во втором – формула (14). Во всех трех процедурах использовались одинаковые гиперпараметры и аугментация данных. Модели обучались в течение 100 эпох с выборкой размера 32. В роли оптимизатора выступал алгоритм Adam со скоростью обучения 0.0001 и нулевым значением параметра d из формулы (9). Параметры β_1 и β_2 из формул (4)–(9) полагались соответственно 0.9 и 0.999. Функция потерь – бинарная кросс-энтропия. Для оценки качества работы нейронной сети вычислялась точность модели в соответствии с формулой (12).

Применялась аугментация данных 2 видов: симметрия относительно вертикальной оси, проходящей через центр изображения, и сдвиги в

горизонтальном и вертикальном направлениях. Для сдвигов устанавливалась максимальная доля от общей ширины или высоты изображения, на которую могла сдвигаться картинка в соответствующем направлении. В процессе обучения величины сдвигов для каждого входного изображения выбирались случайно. В экспериментах использовалось значение 0.4. Появившиеся при сдвигах «пустоты» заполнялись с помощью интерполяционного метода ближайшего соседа.

3.2.1. Классический студент

При обучении классического студента в качестве меток использовались значения 0 или 1, полученные при ручной разметке базы данных. Процесс изменения значений функции потерь и точности во время обучения модели можно увидеть на рисунке 3:

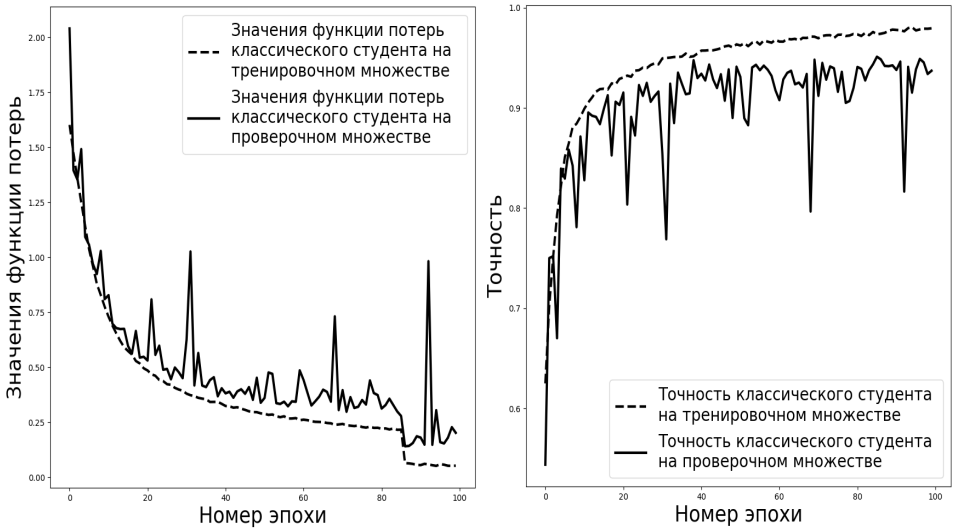

Рис. 3. Динамика значений функции потерь и точности классического студента

Результаты обучения классического студента:

Множество \ Функция		Функция потерь	Точность
Тренировочное множество (лучшее значение в процессе обучения)		0.05183	0.98048
Проверочное множество (лучшее значение в процессе обучения)		0.14111	0.95105
Тестовое множество с ручной разметкой	Без аугментации	0.14394	0.96160
	С аугментацией	0.30252	0.94987

3.2.2. Hard-студент

При обучении данным способом в качестве меток использовались предсказания сети-учителя, распределенные на отрезке [0,1], преобразованные в соответствии с формулой (13). Статистика обучения hard-студента представлена на рисунке 4:

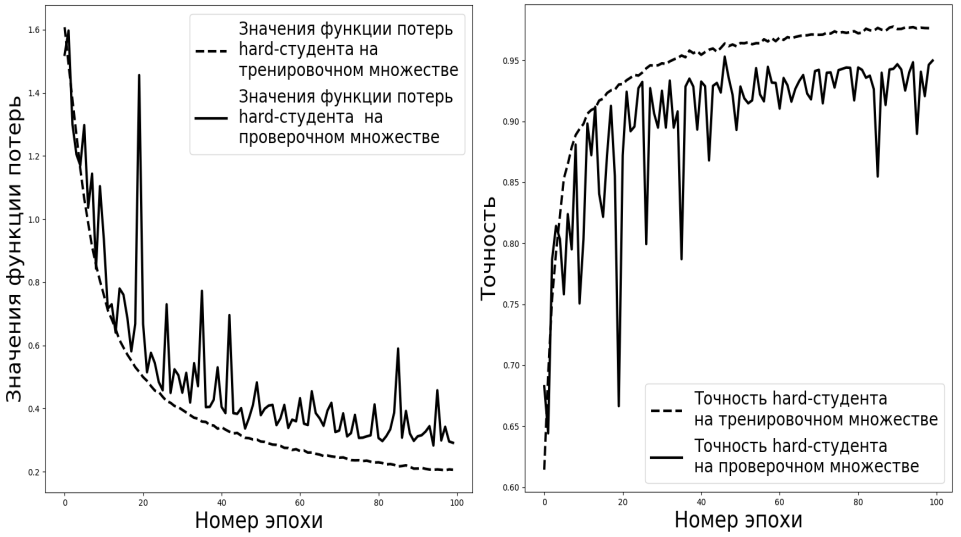

Рис. 4. Динамика значений функции потерь и точности hard-студента

Результаты обучения hard-студента:

Множество \ Функция		Функция потерь	Точность
Тренировочное множество (лучшее значение в процессе обучения)		0.20442	0.97762
Проверочное множество (лучшее значение в процессе обучения)		0.28316	0.95293
Тестовое множество с ручной разметкой	Без аугментации	0.13251	0.95440
	С аугментацией	0.36285	0.94560
Тестовое множество с предсказаниями сети-учителя в качестве меток	Без аугментации	0.12143	0.95493
	С аугментацией	0.35090	0.94667

3.2.3. Soft-студент

В процессе обучения этого студента в роли меток выступали непосредственно предсказания сети-учителя. Таким образом, при обучении модель аппроксимировала не значения $\{0, 1\}$, а действительные числа из отрезка [0,1]. Динамика обучения этой модели показана на рисунке 5:

Результаты обучения soft-студента:

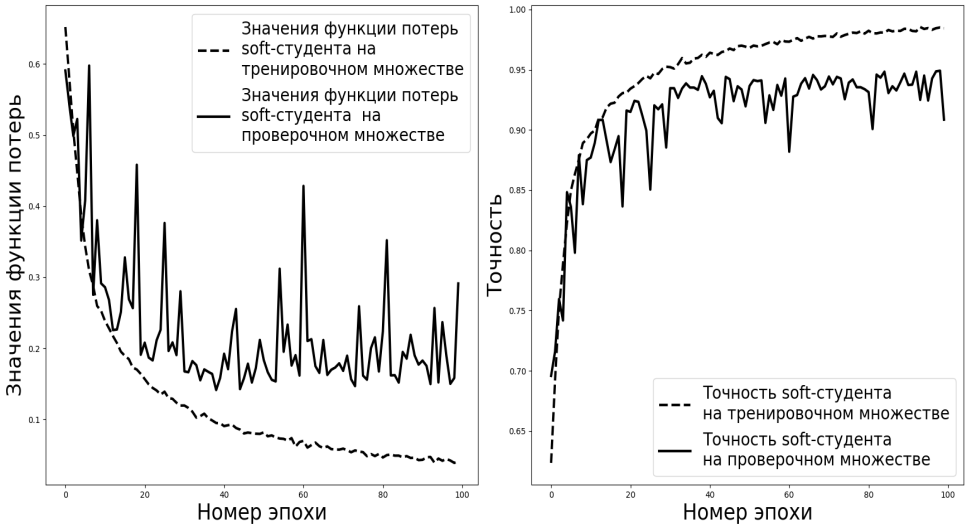

Рис. 5. Динамика значений функции потерь и точности soft-студента

Множество \ Функция		Функция потерь	Точность
Тренировочное множество (лучшее значение в процессе обучения)		0.03890	0.98546
Проверочное множество (лучшее значение в процессе обучения)		0.14147	0.94917
Тестовое множество с ручной разметкой	Без аугментации	0.12629	0.96320
	С аугментацией	0.14906	0.95040
Тестовое множество с предсказаниями сети-учителя в качестве меток	Без аугментации	0.13455	0.96053
	С аугментацией	0.15217	0.95120

4. Исследование обученных нейронных сетей

4.1. Производительность

Среднее время работы нейронных сетей оценивалось на тестовом множестве, состоящем из 3750 изображений. Средний размер изображения составил 0.152 мегапикселя.

Оценка производительности проводилась на устройстве без графического ускорителя с процессором Intel Core i7-7500U, тактовой частотой 2.7 ГГц, 2 ядрами и 16 ГБ оперативной памяти. На этом устройстве были получены следующие результаты:

- Среднее время обработки одного изображения нейронной сетью ResNet-18 – 0.07168 секунды.
- Среднее время обработки одного изображения нейронной сетью ResNet-50 – 0.35452 секунды.
- Общее время обработки тестового множества изображений нейронной сетью ResNet-18 – 268.80433 секунд.
- Общее время обработки тестового множества изображений нейронной сетью ResNet-50 – 1329.45384 секунд.

Таким образом, сеть ResNet-18 в среднем выполняет обработку одного изображения на 0.28284 секунды быстрее модели ResNet-50. А на всём тестовом множестве преимущество составляет 1060.64951 секунд. Получается, что сеть ResNet-18 в среднем работает практически в 5 раз быстрее.

4.2. Статистика распределений предсказаний

Рассмотрим статистику предсказаний всех четырех обученных нейронных сетей на тренировочном, проверочном и тестовом множествах. Сначала построим распределение относительных частот предсказаний моделей, значения которых принадлежат отрезку [0,1] с шагом $h = 0.05$.

1. Тестовое множество. В нем 3750 изображений, по 1875 изображений кошек и собак. Значения относительных частот предсказаний каждой сети приведены в следующих таблицах.

Учитель:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.48027	0.00427	0.0048	0.00293	0.0008
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.00053	0.00187	0.00133	0.00133	0.0008
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.00107	0.00053	0.0008	0.00107	0.00107
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.00187	0.00213	0.0032	0.00347	0.48587

Классический студент:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.4512	0.00987	0.00853	0.00347	0.00267
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.00347	0.00267	0.004	0.00187	0.00267
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.00293	0.00187	0.00293	0.0032	0.00133
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.00213	0.00347	0.0032	0.00773	0.4808

Hard-студент:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.44587	0.01467	0.0072	0.00507	0.00667
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.00373	0.00533	0.00533	0.00427	0.00373
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.00293	0.00453	0.0024	0.00533	0.004
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.00373	0.00347	0.01013	0.01307	0.44853

Soft-студент:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.476	0.00827	0.00293	0.00347	0.00267
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.00267	0.00293	0.00293	0.0032	0.0008
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.00267	0.00187	0.0024	0.0008	0.0024
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.0048	0.00373	0.0048	0.00933	0. 46133

Графическое представление этой статистики можно видеть на рисунке 6:

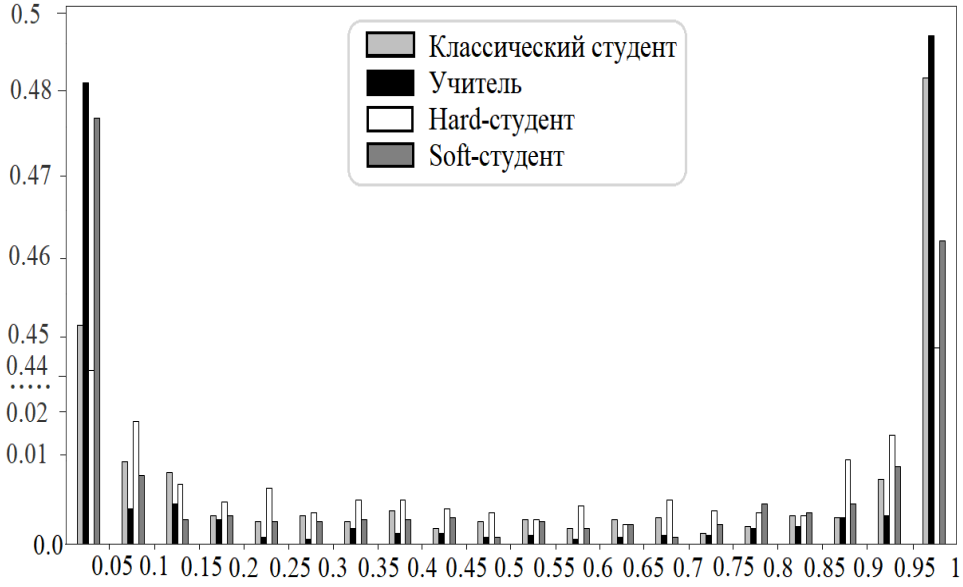

Рис. 6. Распределение относительных частот предсказаний обученных нейронных сетей на тестовом множестве

Заметно, что у всех рассматриваемых сетей в предсказаниях явно преобладают значения очень близкие к 0 или 1, а на остальных проме-

жутках значения во много раз меньше и не превышают 0.015. Можно сказать, что все сети с большой степенью уверенности, то есть очень близко (в данном случае с погрешностью 0.05) к 0 или 1, предсказывают «кошку» или «собаку». Самое уверенное поведение демонстрирует, что вполне ожидаемо, сеть-учитель. Между студентами нельзя выявить наиболее неуверенную сеть – на внутренних полуинтервалах наибольшие значения имеют разные студенты.

2. Проверочное множество. В нем также по 1875 изображений каждого класса. Распределения относительных частот предсказаний для всех моделей изложены в таблицах ниже.

Учитель:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.4808	0.00347	0.00373	0.00107	0.00133
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.00107	0.00187	0.00053	0.0008	0.0024
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.00133	0.00133	0.0008	0.00133	0.00293
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.00267	0.0032	0.00293	0.00373	0.48267

Классический студент:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.45387	0.0104	0.00693	0.00427	0.00507
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.00213	0.0016	0.00373	0.00133	0.00187
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.00213	0.00267	0.0016	0.00213	0.00133
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.00267	0.004	0.0048	0.00773	0.47973

Hard-студент:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.44907	0.01493	0.00933	0.00427	0.00693
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.0048	0.00373	0.004	0.00373	0.0032
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.0024	0.00373	0.00427	0.0048	0.00267
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.00693	0.00587	0.0072	0.01787	0.44027

Soft-студент:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.4752	0.00747	0.00373	0.00453	0.00267
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.0024	0.00187	0.00293	0.00187	0.00213
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.00187	0.0024	0.00293	0.00213	0.0016
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.00533	0.00453	0.00693	0.012	0.45547

Общая картина осталась примерно такой же: как и на тестовом множестве все сети с очень большой степенью уверенности предсказывают или «кошку», или «собаку». А на внутренних промежутках относительные частоты предсказаний не превышают значения 0.015. Студенты стали немного увереннее – большее количество предсказаний стало попадать в промежутки более близкие к краям отрезка $[0, 1]$.

Результаты для проверочного множества изображены на рисунке 7:

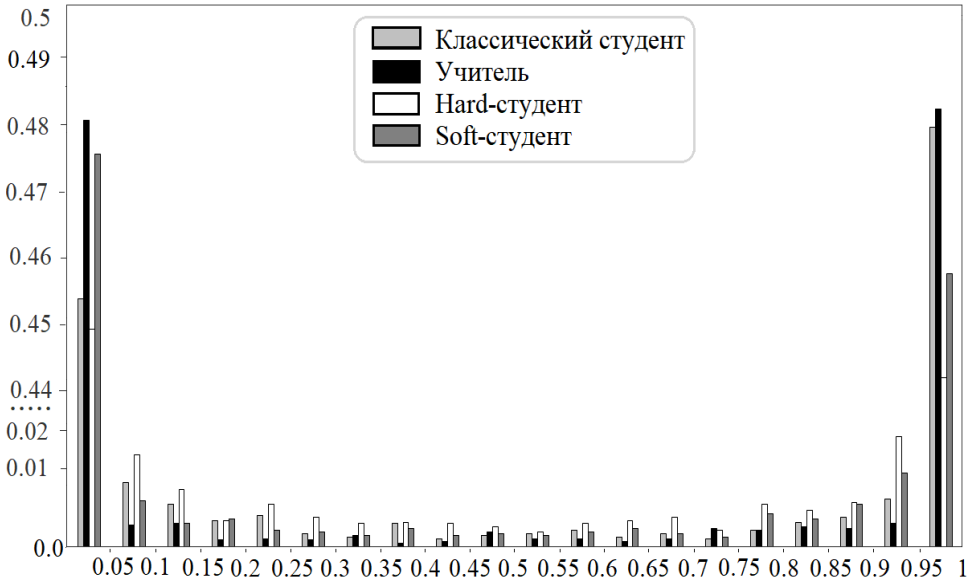

Рис. 7. Распределение относительных частот предсказаний обученных нейронных сетей на проверочном множестве

3. Тренировочное множество. В нем 17500 изображений, по 8750 изображений «кошек» и «собак». Относительные частоты предсказаний обученных моделей представлены в таблицах далее.

Учитель:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.49966	0.00011	0.00006	0.00006	0.0
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.0	0.00006	0.0	0.0	0.0
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.0	0.00006	0.00011	0.00006	0.0
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.00011	0.00011	0.00023	0.00046	0.49891

Классический студент:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.47222	0.00869	0.00451	0.00337	0.00217
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.00149	0.0016	0.00114	0.00131	0.00137
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.00137	0.00126	0.00097	0.00189	0.002
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.002	0.002	0.00291	0.00571	0.48200

Hard-студент:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.46382	0.01274	0.00594	0.00451	0.00406
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.00371	0.00371	0.00229	0.00268	0.00257
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.00229	0.00251	0.00263	0.00291	0.00337
промежуток	[0.75, 0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.00331	0.00451	0.00691	0.01297	0.45257

Soft-студент:

промежуток	[0, 0.05)	[0.05,0.1)	[0.1,0.15)	[0.15,0.2)	[0.2,0.25)
частота	0.49189	0.00406	0.00149	0.00114	0.0008
промежуток	[0.25,0.3)	[0.3, 0.35)	[0.35,0.4)	[0.4,0.45)	[0.45, 0.5)
частота	0.0008	0.00103	0.00086	0.00051	0.0004
промежуток	[0.5,0.55)	[0.55,0.6)	[0.6,0.65)	[0.65,0.7)	[0.7,0.75)
частота	0.00109	0.00126	0.0008	0.0008	0.00131
промежуток	[0.75,0.8)	[0.8,0.85)	[0.85,0.9)	[0.9,0.95)	[0.95,1]
частота	0.00171	0.00251	0.0036	0.00589	0.47806

Результаты для тренировочного множества изображены на рисунке 8:

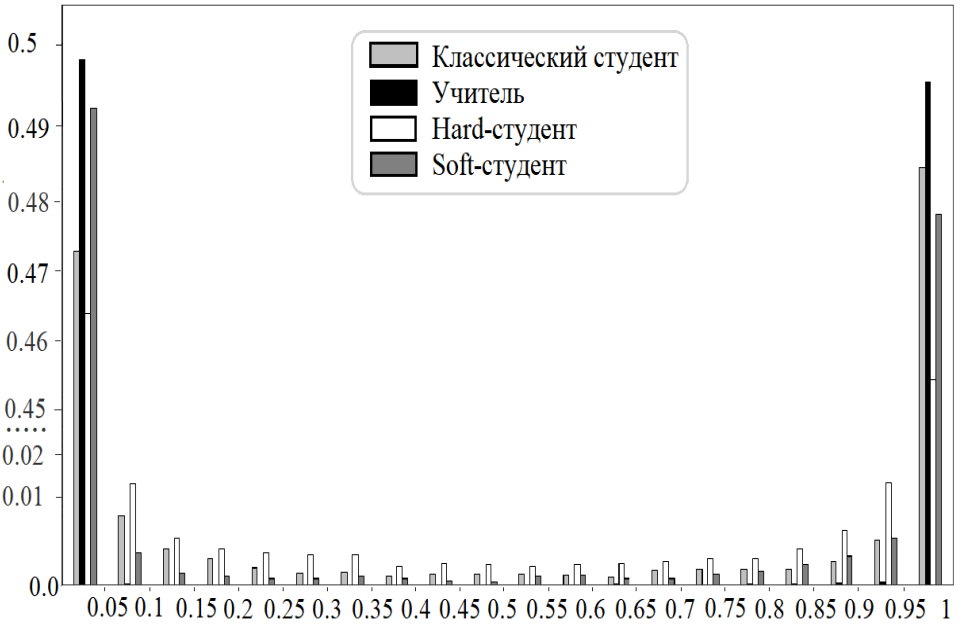

Рис. 8. Распределение относительных частот предсказаний обученных нейронных сетей на тренировочном множестве

Поскольку сети обучались на этом множестве, то и предсказания получаются наиболее уверенными. У каждой сети больше 91% ответов оказываются на промежутках [0,0.05) и [0.95, 1]. У сетей-студентов также в крайние промежутки попадает предсказаний больше, чем на тестовом или проверочном множествах. Видно расслоение графиков – наиболее уверенным является учитель, наименее уверенно ведет себя hard-студент. Промежуточное положение занимают классический студент и soft-студент. Точнее, при предсказаниях менее 0.5 soft-студент везде имеет меньшие значения, чем классический студент, но чем ближе к единице, тем менее уверенным он становится. Возможно, это связано с тем, что soft-студент обучался по ответам сети-учителя. Было выявлено, что сеть-учитель ошибается больше на изображениях с собаками. Ввиду этого, можно сказать, что эта модель более не уверена в их отношении, что естественно отражается в предсказаниях, выступивших в качестве меток при обучении soft-студента. Тем самым сеть-учитель может передавать свою неуверенность студенту.

4.3. Степень неуверенности на множестве

Для задачи бинарной классификации введем понятие «степень неуверенности модели на множестве», вычисляемая по следующей формуле:

$$unc = \frac{1}{n}\sum_{i=1}^{n}(p_i - y_i)^2 \ , \tag{15}$$

где n – мощность множества, на котором вычисляется степень неуверенности, y_i – значение, принимаемое за ответ модели на этом объекте, p_i – предсказание модели на i-ом изображении этого множества. В рассматриваемой задаче за ответ модели на i-ом объекте принимается предсказание сети, преобразованное в соответствии с формулой (13), то есть $y_i \in \{0, 1\} \, \forall \, i = \overline{1, n}$.

В следующей таблице приведены степень неуверенности и точность для каждой из рассматриваемых нейронных сетей на тренировочном, проверочном и тестовом множествах (с ручной разметкой). В каждой клетке таблицы выше диагонали указана точность модели на соответствующем столбцу множестве. А ниже диагонали – степень неуверенности, посчитанная по формуле (15).

Сеть \ Множество	Тренировочное множество	Проверочное множество	Тестовое множество
Учитель	0.99983 / 0.00006	0.97526 / 0.00252	0.97333 / 0.00197
Классический студент	0.98048 / 0.00262	0.95105 / 0.00402	0.96160 / 0.00467
Hard-студент	0.97762 / 0.005	0.95293 / 0.00674	0.95440 / 0.00713
Soft-студент	0.98546 / 0.00163	0.94917 / 0.00421	0.96320 / 0.00403

4.4. Рекурсивное применение технологии Knowledge Distillation

Был дважды проведен эксперимент по рекурсивному обучению сетей ResNet-18 по технологии Knowledge Distillation. В каждом испытании

на новом шаге в качестве меток выступали предсказания сети ResNet-18, полученной на предыдущем шаге. Эти предсказания предварительно преобразовывались по формулам (13) и (14), на первом шаге использовались предсказания hard-студента и soft-студента в первом и втором экспериментах соответственно. При обучении всех моделей для чистоты эксперимента использовались те же гиперпараметры обучения, как и при обучении сетей-студентов. Таким образом, был использован оптимизатор Adam со скоростью обучения 0.0001 и нулевым значением параметра d из формулы (9). Параметры β_1 и β_2 из формул (4)–(9) равнялись 0.9 и 0.999 соответственно. Для оценки качества работы моделей вычислялась точность сети в соответствии с формулой (12). Функция потерь – бинарная кросс-энтропия. Каждая модель обучалась 100 эпох с выборкой размера 32. При рекурсивном обучении применялась аугментация данных, как и при обучении сетей-студентов.

В следующих таблицах для каждого эксперимента приведены точности и значения степени неуверенности на всех рассматриваемых множествах для сетей, полученных в ходе этих испытаний. В каждой клетке таблицы выше диагонали указана точность, а ниже – степень неуверенности модели на соответствующем множестве.

Результаты рекурсивного обучения с помощью hard-студента:

Множество / Сеть	Тренировочное множество	Проверочное множество	Тестовое множество
Hard-студент	0.97762 0.005	0.95293 0.00674	0.95440 0.00713
ResNet-18 №1	0.98225 0.00391	0.94621 0.0069	0.95760 0.006
ResNet-18 №2	0.97463 0.00379	0.94567 0.00569	0.94880 0.00515
ResNet-18 №3	0.98025 0.00292	0.94809 0.0062	0.94773 0.00668
ResNet-18 №4	0.98117 0.0021	0.94567 0.00556	0.94293 0.00575
ResNet-18 №5	0.96336 0.00616	0.94621 0.00866	0.93600 0.00895
ResNet-18 №6	0.97281 0.00352	0.94594 0.00572	0.93520 0.00561

Результаты рекурсивного обучения с помощью soft-студента:

Сеть \ Множество	Тренировочное множество	Проверочное множество	Тестовое множество
Soft-студент	0.98546 0.00163	0.94917 0.00421	0.96320 0.00403
ResNet-18 №1	0.98397 0.00268	0.95535 0.00529	0.96320 0.00454
ResNet-18 №2	0.98363 0.00636	0.95697 0.00934	0.95733 0.00935
ResNet-18 №3	0.97933 0.00937	0.95966 0.01236	0.95093 0.01272
ResNet-18 №4	0.97670 0.01249	0.95858 0.01547	0.95307 0.01576
ResNet-18 №5	0.97384 0.01433	0.95320 0.01795	0.94507 0.01856
ResNet-18 №6	0.97258 0.01940	0.95643 0.02302	0.94587 0.02268

В обоих экспериментах наблюдается относительное снижение точности в процессе рекурсивного обучения, но и в том, и в другом случаях небольшое – менее 3%. Однако в отношении степени неуверенности в этих экспериментах получены различные результаты. В первом случае значение степени неуверенности на каждом множестве практически не меняется на всех шагах – отличие от hard-студента не более 0.003. Во втором эксперименте, наоборот, наблюдается явный рост значений этой величины на каждом множестве. При этом точность на тестовом множестве у сети, обученной в ходе эксперимента с soft-студентом, всегда больше, чем у модели, полученной на аналогичном шаге в опыте с hard-студентом.

Кроме того, для рекурсивно обученных сетей были изучены распределения относительных частот предсказаний на тестовом множестве. Для первого эксперимента, поскольку значения степени неуверенности не увеличивались и оставались очень малыми, то и распределения предсказаний остались примерно такими же, как и для hard-студента. Напротив, в ходе эксперимента с soft-студентом с каждым новым шагом всё большее

количество предсказаний попадало во внутренние полуинтервалы за счет уменьшения значений в крайних промежутках. В большей степени эта закономерность проявилась на отрезке $[0.5,\ 1]$, то есть на промежутках, попадание в которые интерпретируется как принадлежность объекта к классу «изображение с собакой». Эту динамику можно увидеть на рисунке 9:

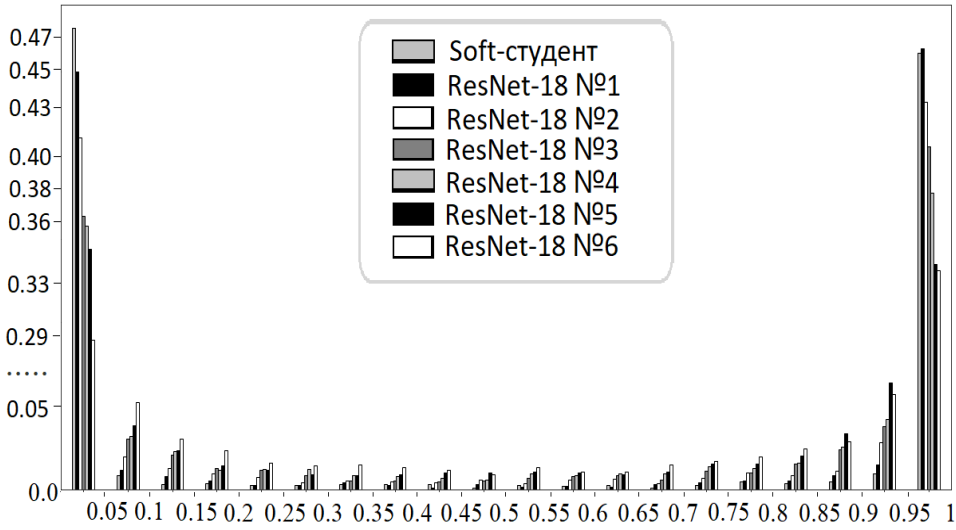

Рис. 9. Распределение относительных частот предсказаний на тестовом множестве рекурсивно обученных с помощью soft-студента нейронных сетей

Таким образом, помимо общего увеличения значения степени неуверенности у рекурсивно обученных с помощью soft-студента сетей, наблюдается наследование характера распределения этой неуверенности. Тем самым, получает подтверждение предположение о возможности наследования неуверенности обучаемой сетью от своей сети-учителя.

5. Выводы

В ходе данной работы подтверждена возможность обучения по технологии Knowledge Distillation несложных моделей ResNet-18, способных выдавать результаты, сопоставимые с аналогичными измерениями для сети-учителя, полученной из нейронной сети ResNet-50, для задачи бинарной классификации. При этом использование сети-студента позво-

ляет увеличить производительность вычислений по сравнению с сетью-учителем практически в 5 раз.

Все три сети-студента дают примерно одинаковые результаты, причем, по сравнению с классическим студентом, soft-студент показал лучший результат на тестовом множестве с ручной разметкой, как с применением аугментации данных, так и без неё. Hard-студент, в свою очередь, продемонстрировал результаты ненамного хуже классического студента, но соответствующие значения параметров были получены практически в два раза быстрее: после 86 эпох – у классического и после 47 эпох – у hard-студента.

Были изучены распределения относительных частот предсказаний на различных множествах для каждой обученной нейронной сети. Замечена закономерность между характером ошибок сети-учителя и soft-студента, проявившаяся в переносе неуверенности сети-учителя в предсказании принадлежности изображения к классу «изображение с собакой» на аналогичное поведение, выявленное у soft-студента. У hard-студента это влияние не столь заметно, поскольку детали подобного рода нивелируются в форме представления предсказаний сети-учителя, выступавших в качестве меток в процессе обучения этой модели, в отличие от soft-студента, который в процессе обучения аппроксимировал непосредственно предсказания сети-учителя. Такая же тенденция выявлена при рекурсивном обучении моделей по данной технологии.

В процессе изучения распределений относительных частот предсказаний было введено понятие степени неуверенности модели на множестве как величины отклонения предсказаний нейронной сети от значений, принимаемых за ответ модели на рассматриваемом множестве. Эту величину можно рассматривать как интуитивно понятную меру способности модели к обобщению. На всех обученных сетях значение степени неуверенности очень небольшое, что может означать, что способность к обобщению у моделей, обученных по технологии Knowledge Distillation, сохраняется.

Также рекурсивно были обучены модели ResNet-18 по технологии Knowledge Distillation, с обеими формами представления предсказаний сети-учителя. Подобная процедура в обоих случаях показала вполне стабильные результаты: обученные в ходе каждого эксперимента модели делают достаточно точные предсказания на всех множествах, выступая наравне с сетями-студентами. Более того были получены нейронные сети, которые не уступают на тестовом множестве моделям, обученным с помощью более сложной нейронной сети.

Дальнейшие исследования будут направлены на изучение возможности применения данной технологии для других задач.

Автор выражает благодарность доценту кафедры МаТИС механико-математического факультета МГУ, к.ф.-м.н. Часовских Анатолию Александровичу и доценту кафедры высшей математики института Кибернетики РТУ МИРЭА, к.т.н. Парфенову Денису Васильевичу за постановку задачи и помощь в исследовании.

Список литературы

[1] Earnest Paul Ijjina, Dhananjai Chand, Savyasachi Gupta, Goutham K, *Computer Vision-based Accident Detection in Traffic Surveillance*, arXiv: arxiv.org/abs/1911.10037.

[2] Omid Bazgir, Ruibo Zhang, Saugato Rahman Dhruba, Raziur Rahman, Souparno Ghosh, Ranadip Pal, *REFINED (Representation of Features as Images with Neighborhood Dependencies): A novel feature representation for Convolutional Neural Networks*, arXiv: arxiv.org/abs/1912.05687.

[3] Sara Sabour, Nicholas Frosst, Geoffrey E Hinton, *Dynamic Routing Between Capsules*, arXiv: arxiv.org/abs/1710.09829.

[4] Jo Schlemper, Chen Qin, Jinming Duan, Ronald M. Summers, Kerstin Hammernik, $\sum$*-net: Ensembled Iterative Deep Neural Networks for Accelerated Parallel MR Image Reconstruction*, arXiv: arxiv.org/abs/1912.05480.

[5] Ramit Pahwa, Manoj Ghuhan Arivazhagan, Ankur Garg, Siddarth Krishnamoorthy, Rohit Saxena, Sunav Choudhary, *Data-Driven Compression of Convolutional Neural Networks*, arXiv: arxiv.org/abs/1911.12740.

[6] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, *Distilling the Knowledge in a Neural Network*, arXiv: arxiv.org/abs/1503.02531.

[7] Gaurav Menghani, Sujith Ravi, *Learning from a Teacher using Unlabeled Data*, arXiv: arxiv.org/abs/1911.05275.

[8] Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun, *Deep Residual Learning for Image Recognition*, arXiv: arxiv.org/abs/1512.03385.

[9] Система организации конкурсов по исследованию данных «Kaggle», *Соревнование «Dogs vs. Cats», www.kaggle.com/c/dogs-vs-cats/overview.*

[10] Саймон Хайкин, *Нейронные сети: полный курс*, 2-е издание, Издательский дом «Вильямс», Москва, 2006, 1104 с.

[11] George Philipp, Dawn Song, Jaime G. Carbonell, *The exploding gradient problem demystified – definition, prevalence, impact, origin, tradeoffs, and solutions*, arXiv: arxiv.org/abs/1712.05577.

[12] Ян Эрик Солем, *Программирование компьютерного зрения на языке Python*, ДМК Пресс, Москва, 2016, 312 с.

[13] В.Б.Кудрявцев, Э.Э. Гасанов, А.С. Подколзин, *Основы теории интеллектуальных систем*, МАКС Пресс, Москва, 2016, 612 с.

[14] Bao-Gang Hu, Ran He, XiaoTong Yuan, *Information-Theoretic Measures for Objective Evaluation of Classifications*, arXiv: arxiv.org/abs/1107.1837.

[15] Mitchell A. Gordon, Kevin Duh, *Explaining Sequence-Level Knowledge Distillation as Data-Augmentation for Neural Machine Translation*, arXiv: arxiv.org/abs/1912.03334.

[16] Tyler Sypherd, Mario Diaz, Lalitha Sankar, Peter Kairouz, *A Tunable Loss Function for Binary Classification*, arXiv: arxiv.org/abs/1902.04639.

[17] Diederik P. Kingma, Jimmy Ba, *Adam: A Method for Stochastic Optimization*, arXiv: arxiv.org/abs/1412.6980.

The technology of knowledge distillation for training neural networks on example of binary classification

Biryukova V.A.

Using the technology of training neural networks, the knowledge distillation, models were obtained that solve the binary classification problem with the productivity that is about five times higher than the performance of the teacher network with an insignificant drop in quality. The convolutional neural network ResNet-18 was trained in two ways by this technology (using the pre-trained network ResNet-50) and by the classical method. The concept of the degree of uncertainty of the model on objects' set is introduced as the quantity of the deviation of the neural network predictions from the values accepted for the answer. The experiments on the recursive application of the knowledge distillation technology were also conducted.

Keywords: knowledge distillation, binary classification, residual neural network, convolutional neural network, degree of uncertainty of the model on objects' set, recursive training of neural networks.