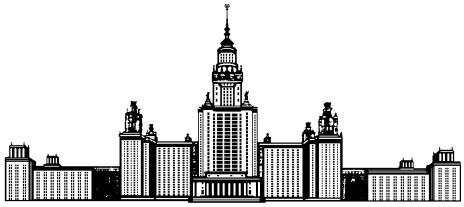

Московский государственный университет имени М. В. Ломоносова

Факультет вычислительной математики и кибернетики

Кафедра суперкомпьютеров и квантовой информатики

Камалов Ирек Маратович

Исследование методов решения задачи обучения по нескольким примерам, основанных на применении ансамбля алгоритмов формирования синтетических обучающих данных

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:

к.ф-м.н., ассистент

Д.Ю.Буряк

Москва, 2021

Содержание

1 Введение

Многие прикладные задачи в области глубокого обучения предполагают работу с меняющимися классами [1][2][4]. Например, такие задачи часто возникают в робототехнике [1][2][3], когда требуется быстрая адаптация к новым локациям, новым объектам. Также необходимость в адаптирующихся моделях возникает и за пределами компьютерного зрения. Так одной из наиболее актуальных задач в обработке естественных языков [4] является генерация текста по нескольким входным предложениям, раскрывающим его смысл.

В приведённых работах адаптация моделей к меняющимся задачам сводится к вопросу их обучения по небольшому числу примеров. При этом классические методы их построения с помощью глубоких сетей приводят к трудностям в реализации, связанным с

1) необходимостью обеспечить большое количество размеченных примеров

2) вычислительной сложностью обучения [5]

Данные факторы приводят к высокой актуальности алгоритмов, предлагающих эффективные способы построения моделей по небольшой выборке.

Успех генеративных сетей для синтеза изображений и речи [6][7][8] привёл к попыткам использовать их для осуществления аугментации в задачах распознавания [9][10]. Данная идея нашла применение в методах обучения по небольшому числу примеров, стремящихся воссоздать неизвестную обучающую выборку. При наличии репрезентативной искусственной выборки задача может быть сведена к классическим методам обучения.

Данная работа посвящена изучению и развитию существующих методов генерации синтетических выборок, решающих задачу классификации по нескольким примерам. Целью работы является анализ наиболее эффективных алгоритмов и их усовершенствование посредством применения в ансамбле.

1.1 Формальное описание задачи.

Рассматривается задача классификации изображений с учителем.

Определение. *Эпизодом или целевой задачей классификации изображений по нескольким примерам будем называть задачу, для которой:*

1. *На вход подается обучающая выборка, состоящая из нескольких непересекающихся классов, для каждого из которых дано небольшое количество размеченных изображений*

2. *Требуется построить алгоритм $h : X \to Y$ отображающий множество объектов классов во множество меток*

Целевыми примерами будем называть изображения из небольшой входной выборки или любые их описания, нецелевыми, соответственно, изображения любых других классов или их описания. Для решения задачи допускается использование примеров нецелевых классов. На их количество ограничений не накладывается.

Стоит отметить, что методы машинного обучения, базирующиеся на оптимизации эмпирического риска, для решения данной задачи не подходят, поскольку при недостатке обучающих примеров приводят к переобучению [11]. Это связано с тем, что эмпирический риск даёт качественную оценку теоретического риска лишь в случаях, когда обучающая выборка имеет достаточно большой размер [11]. Различные алгоритмы предлагают оригинальные способы борьбы с этой проблемой [11].

2 Обзор литературы

Для построения моделей по небольшому объёму данных в большинстве работ предлагаются алгоритмы, обучение которых происходит также на нецелевых примерах, доступных в относительно большем количестве [11]. Предполагается, что это позволит каким-то образом перенести знания о закономерностях в других данных на целевую задачу. В зависимости от алгоритмов обучения, применяемых к нецелевым примерам, в литературе делят подходы на 3 типа: модельный, алгоритмический и генеративный [11]. Этап обучения, работающий с объектами нецелевых классов в литературе часто называют мета-обучением [12][13][11], так как его семантика связана с поиском закономерностей общего характера, не связанных с фиксированной задачей.

Также стоит отметить, что многие методы в этой области предполагают переход от анализа изображений к работе с векторами признаков [13][14][17]. Строго говоря, переход от изображения к его вектору признаков является понижением размерности тензора с наименьшими потерями. Качество понижения может быть как эвристикой в рамках выбранной задачи [17], так и предметом обучения [13][14]. В качестве основного приёма перевода изображения в вектор используется расчёт выхода одного из слоёв глубокой сети [13][14][17]. В некоторых случаях глубокая сеть предварительно проходит тонкую настройку [17]. Этот приём имеет экспериментальные обоснования [15] и успешно используется, например, в задаче локализации [16]. Далее в работе будут неоднократные отсылки к данному приёму.

2.1 Модельный подход.

В рамках данной группы методов для решения вводится некий набор параметров, предположительно не зависящих от целевой задачи. Например, в работе [13] в качестве таких параметров рассматриваются веса глубоких свёрточных сетей, с помощью которых изображения преобразуются в векторы. Рассматриваются 2 нейронные сети g, f, с помощью которых конструируется составная архитектура. g преобразует в векторы признаков изображения из небольшой обучающей выборки, f преобразует тестовое изображение для распознавания. Вероятность принадлежности входного объекта z i-му классу в простом случае, когда для класса i есть лишь 1 обучающий пример x_i, считается как $a_i(z, x_i) = softmax(c(f(z), g(x_i))) = \frac{e^{c(f(z),g(x_i))}}{\sum\limits_j e^{c(f(z),g(x_j))}}$, где $c(x, y)$ некая метрика расстояния. Модель целевого классификатора строится из описанных компонент, включая нейронные сети f, g и расчёт вероятностей a_i (рис.1). Заметим, что по полученной архитектуре может быть составлен граф вычислений для применения алгоритма обратного распространения ошибки.

Как и во многих работах [11], для подготовки целевого классификатора производится мета-обучение. В его рамках формируются входные эпизоды (L, S), имитирующие целевые задачи, но составленные из объектов нецелевых классов. L - набор меток, S - небольшое множество обучающих примеров. Также формируется пакет тестовых изображений B, который необходимо научиться распознавать по (L, S). Стохастическая оптимизация состоит в минимизации ошибки классификации по таким пакетам данных (L, S, B). Функция ошибки - перекрёстная энтропия $loss = CrossEntropy(a, y) = -\sum\limits_{i=1}^{N} y_i \log_2 a_i$, где N - количество классов, a - набор вероятностей, предсказанный моделью, y - вектор истинных вероятностей из нулей и одной единицы. Предполагается, что с помощью такого алгоритма можно найти некоторые параметры модели, при которых качество классификации для произвольной задачи (L, S, B) будет в среднем наилучшим. Поскольку такое обучение не привязано к работе с фиксированными классами, его относят к алгоритмам мета-обучения.

2.2 Алгоритмический подход.

Методы данного подхода отличаются от предыдущих тем, что параметры модели, введённой для решения задачи, не являются общими для всех целевых эпизодов. Например, в работе [12] строится алгоритм поиска весов классификатора, который проходит стадию мета-обучения на нецелевых данных, но на каждой новой задаче будет

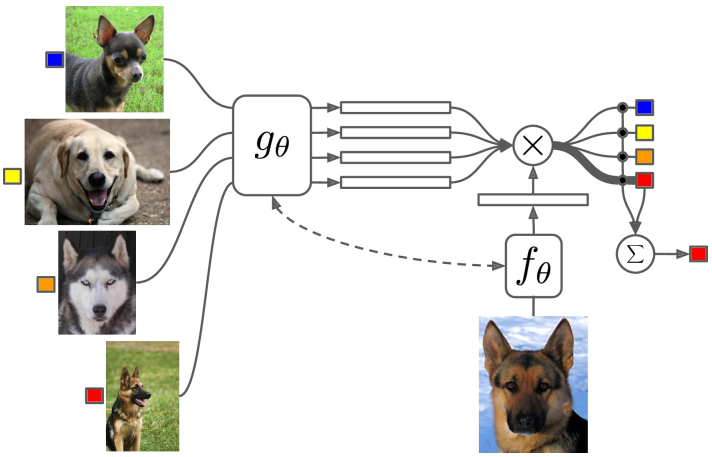

Рис. 1: Пример архитектуры модельного алгоритма.

дополнительно обучаться.

Так же, как в рассмотренной работе [13], для обучения формируются небольшие выборки из нецелевых примеров, имитирующие тестовые задачи. Стохастическая оптимизация производится в два этапа: для каждой выборки из пакета $\{\tau_i\}, i = \overline{1,n}$ вычисляются веса θ_i' после нескольких шагов градиентного спуска, настраивающего классификатор f_θ на целевую задачу τ_i, но веса модели изменяются лишь на втором шаге, который совершается в сторону антиградиента от суммы ошибок классификаторов, претерпевших несколько шагов градиентного спуска $\theta \leftarrow \theta - \beta\nabla_\theta \sum\limits_i L(f_{\theta_i'}(\tau_i))$.

Каждая итерация данного обучения стремится в среднем минимизировать ошибку классификатора $f_{\theta'}$, который потенциально получится в результате настройки весов на некоторую входную задачу τ. Далее при возникновении задачи, будут сделаны несколько шагов градиентного спуска. Вновь наблюдается этап предварительного обучения, не привязанного к работе с фиксированным множеством классов.

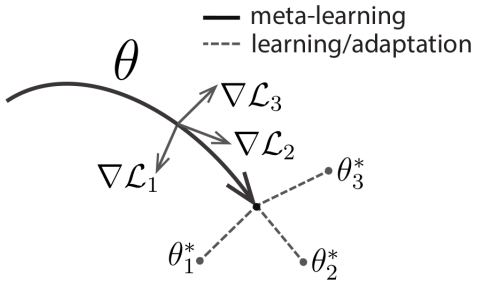

Рис. 2: Мета-обучение градиентным спуском.

2.3 Генеративный подход.

Существуют также алгоритмы расширения входной выборки [17][18], с помощью которых задачу можно свести к классическим методам машинного обучения. Построив алгоритм генерации искусственных примеров, принимающий на вход небольшое количество изображений, можно воссоздать неизвестную обучающую выборку. На полученной выборке может быть обучен целевой классификатор. Далее в работе будет подробно рассматриваться данный класс методов.

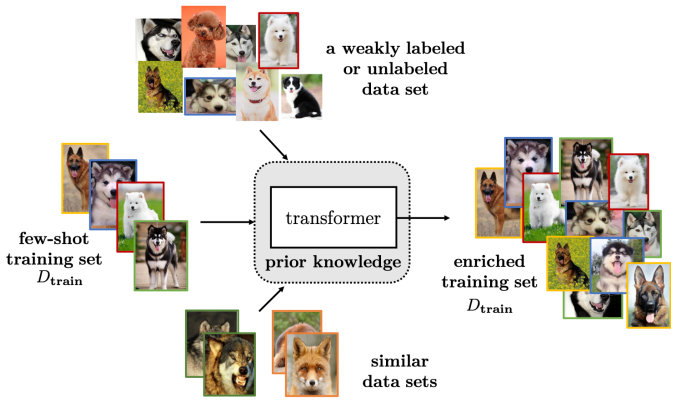

Рис. 3: Генерация выборки по небольшому числу изображений.

3 Постановка задачи.

В рамках работы была поставлена задача реализовать следующие подпункты.

- Провести обзор существующих методов генерации искусственных выборок и отобрать перспективные алгоритмы, которые могут быть применены совместно.
- Предложить стратегию ансамблирования генераторов, позволяющую создавать более репрезентативную выборку, в сравнении с отдельными методами.
- Реализовать предлагаемое решение с выбранными алгоритмами генерации.
- Провести тестирование ансамбля вместе с целевой моделью на разных базах изображений.
- На основании тестирования провести исследование точности классификатора, обучаемого на выборке, сгенерированной ансамблем.

4 Решение задачи с помощью ансамбля алгоритмов генерации.

Из практики построения нейронных сетей можно выделить методы аугментации изображений, позволяющие увеличить количество обучающих примеров, включающие аффинные преобразования, зашумление, растяжение, обрезку изображений и т.п. На практике такие методы редко используются для построения выборки в рамках данной задачи [11].

Ключевой идеей при построении алгоритмов генерации, повлиявшей на большинство предлагаемых методов, является идея поиска закономерностей, связывающих пары примеров одного класса [11][17][18]. Далее будут рассмотрены 2 эффективных алгоритма генерации, построенные по этому принципу [17][18]. Оба метода вместо изображений принимают на вход векторы длины S, полученные в результате преобразования тензоров предобученной свёрточной сетью, и в качестве примеров генерируют векторы той же длины. Данные векторы принадлежат некому пространству $\mathbb{R}^S$.

4.1 Дельта-кодировщик.

Определение. *$\triangle$-кодировщиком будем называть нейронную сеть с архитектурой, удовлетворяющей свойствам:*

1) Размерность входного слоя сети составляет 2S

2) Одним из скрытых слоев является полносвязный слой размерности l, много меньшей по сравнению с величиной S

3) Слой, идущий следом за данным, имеет размерность S + l

4) Размерность выходного слоя равна S

Подсеть $\triangle$-кодировщика, находящаяся в сети до слоя размера l, называется кодировщиком, а идущая следом - декодировщиком.

Обучение данной модели происходит на случайных парах нецелевых примеров. Пара X^S, Y^S подается на вход первой подсети, после чего та рассчитывает вектор Z небольшой размерности l. Далее вторая подсеть осуществляет регрессию одного из входных примеров X^S, получая на вход Z, Y^S. Результирующий вектор - $\widehat{X^S}$, функция ошибки $loss = weightedMAE(\widehat{X^S}, X^S) = \sum\limits_i w_i|\widehat{X^S}_i - X^S_i|$, где веса $w_i = |\widehat{X^S}_i - X^S_i|/||\widehat{X^S} - X^S||$ вводятся для избежания больших градиентов. Предполагается, что если размер вектора Z достаточно мал, то при обучении первая подсеть не сможет кодировать в нём

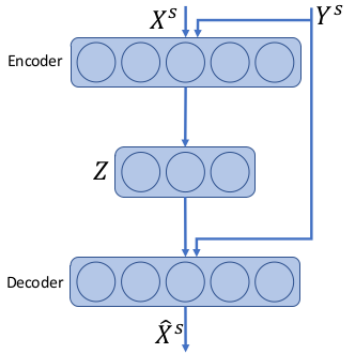

Рис. 4: Обучение △-кодировщика.

непосредственно X^S, чтобы вторая подсеть восстанавливала X^S по Z, поэтому вся сеть будет вынуждена кодировать некую информацию, связывающую векторы X^S, Y^S, чтобы с её помощью более эффективно воссоздавать X^S, трансформируя Y^S.

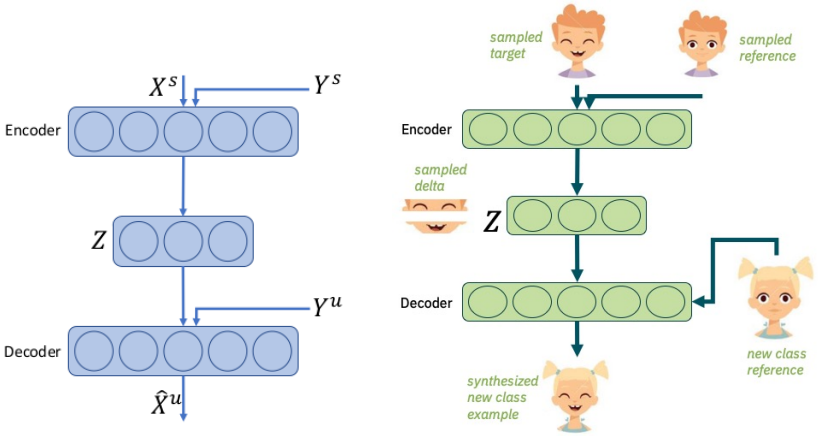

Рис. 5: Генерация примеров △-кодировщиком.

Примеры новых классов генерируются △-кодировщиком путём преобразования векторов Y^u из небольшой входной выборки. Обученная сеть трансформирует входной пример на основании произвольной пары нецелевого класса X^S, Y^S, которая подается на вход первой подсети. Предполагается, что информация об их связи, оказавшаяся в векторе Z, в результате обучения не зависит от класса X^S, Y^S, и применение второй подсети к вектору Y^u преобразует его в пример нового класса $\widehat{X^u}$, как преобразовало бы Y^S в X^S, если бы на вход декодировщика подавался Y^S.

4.2 Центроидный генератор.

Как и в случае △-кодировщика все тензоры изображений преобразуются в векторы пространства $\mathbb{R}^S$ с помощью глубокой свёрточной сети. Каждое множество векторов

нецелевого класса кластеризуется методом k-средних. Далее алгоритм работает лишь с центроидами полученных кластеров. Центроиды разбиваются на четверки векторов следующим образом: четвёрка составляется из 2 пар (C_1a, C_2a), (C_1b, C_2b). Каждая пара принадлежит одному классу, при этом среди четвёрок выбираются те, для которых косинусное расстояние $\rho_{cos}(C_1a - C_2a, C_1b - C_2b)$ имеет меньшее значение. Четвёрки векторов, для которых данное расстояние действительно очень мало, образуют пары, в которых прослеживаются закономерности, связывающие перекрёстные векторы C_1a, C_1b и C_2a, C_2b. Это означает, что пары (C_1a, C_2a), (C_1b, C_2b) имеют некоторое сходство, и прослеживается аналогия в том, по каким критериям различаются между собой центроиды внутри пар. По этой причине в оригинальной работе четвёрки векторов также называются аналогиями. На рисунке 6 представлены 2 примера с такими аналогиями.

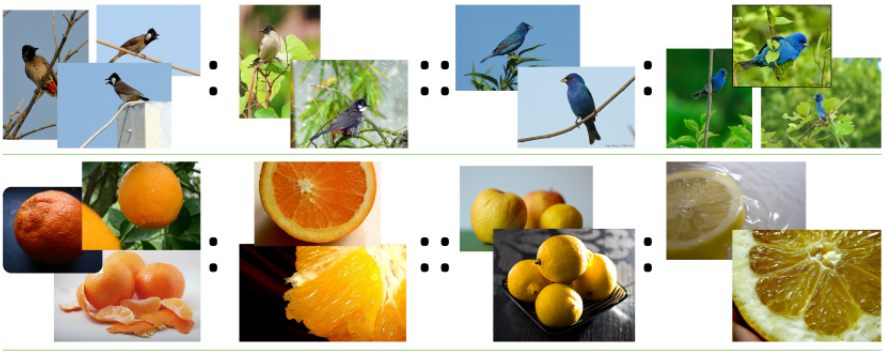

Рис. 6: Четвёрки с малым косинусным расстоянием. **Строка1:** Птицы на фоне неба и на фоне зелени. **Строка2:** Нетронутые цитрусы и порезанные цитрусы. В обоих случаях птицы и цитрусы не принадлежат одному классу для разных пар.

Определение. *Центроидным генератором* $G(C_1a, C_2a, C_1b)$ *будем называть полносвязную нейронную сеть с архитектурой, удовлетворяющей свойствам:*

1) Размерность входного слоя сети составляет $3S$

4) Размерность выходного слоя равна S

При обучении по указанным четвёркам на вход сети подаются 3 центроиды C_1a, C_2a, C_1b, где (C_1a, C_2a) принадлежат классу a, C_1b классу b. Выходной вектор обозначим как $\widehat{C_2b}$. Функция ошибки $loss = MSE(\widehat{C_2b}, C_2b) = \frac{1}{N}\sum_{i}^{N}(\widehat{C_2b}_i - C_2b_i)^2$. Таким образом, модель обучается регрессии центроиды C_2b, парной для C_1b. Предполагается, что для достижения лучшего качества сеть будет вынуждена использовать закономерности, заложенные в аналогиях между парами.

Генерация примеров происходит через трансформацию входных объектов X из небольшой обучающей выборки. Синтетические примеры, соответствующие неизвестным цен-

троидам целевых классов получаются как $Y = G(C_1a, C_2a, X)$ путём попытки переноса геометрических аналогий, распознанных сетью на этапе обучения на новую пару (X, Y). (C_1a, C_2a) варьируются в ходе генерации.

4.3 Ансамбль алгоритмов генерации.

В качество нового подхода к решению задачи предлагается алгоритм ансамблирования методов генерации, состоящий из этапов:

1. Перевод всех тензоров изображений в векторы признаков.
2. Обучение нескольких генераторов на примерах нецелевых классов.
3. Генерация обучающей выборки каждым генератором по отдельности.
4. Смешивание полученных выборок в некотором соотношении, которое является гиперпараметром ансамбля.
5. Обучение целевой модели на смешанной выборке.

В качестве генераторов в работе рассматриваются $\triangle$-кодировщик и центроидный генератор. Этапы решения задачи классификации можно описать схемой на рисунке 7. Подготовка классификатора разбита на 2 этапа. В первом проходит обучение вспомогательных моделей - глубокой сети и двух генераторов. Глубокая сеть обучается переводу тензоров изображений в векторы признаков. Данные векторы являются её выходами и подаются на вход генераторов в ходе их обучения и в ходе генерации. Второй этап включает построение 2 выборок, смешивание и обучение классификатора. При построении выборок генераторы используют векторы целевых и нецелевых примеров. На схеме два этапа разделены вертикальной чертой.

5 Вычислительные эксперименты

5.1 Программная реализация.

Для реализации глубокой сети, генераторов и целевого классификатора используется язык Python 3.7 и библиотека PyTorch с поддержкой вычислений на графическом процессоре. Обучение сетей производится с помощью ускорителя NVIDIA Tesla V100. В библиотеке PyTorch были использованы подмодули torch и torchvision.

Два генератора и целевая модель были реализованы как наследники базового класса nn.Module библиотеки torch и обучены самостоятельно. Были заимствованы функции

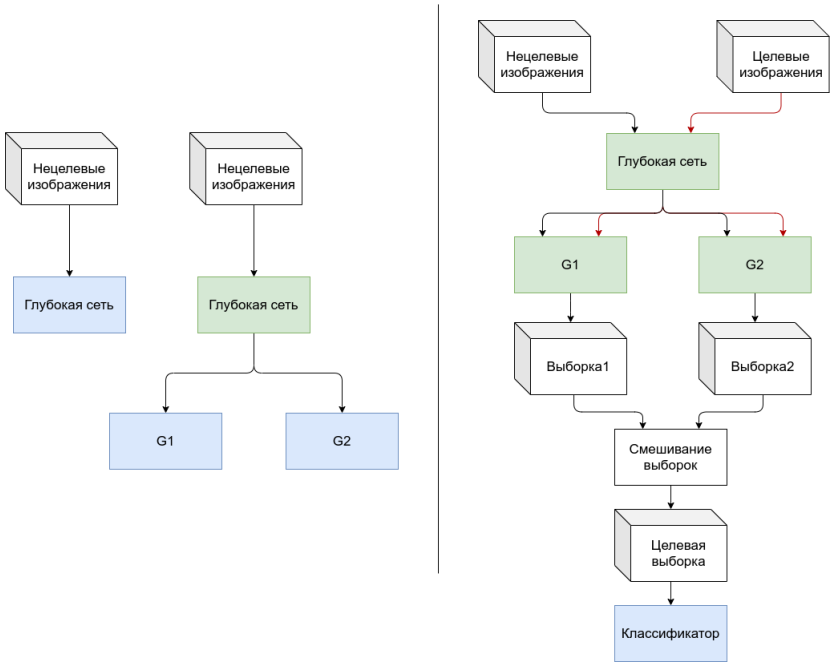

Рис. 7: Общая схема предлагаемого решения. Синие блоки - обучение моделей, зелёные - использование. G1 - центроидный генератор, G2 - △-кодировщик. Стрелки из зелёных блоков глубокой сети означают использование их выходов в качестве входных векторов.

предобработки векторов для разбиения их на пары при обучении △-кодировщика и на четвёрки при обучении центроидного генератора.

Для реализации глубокой сети, как в работе [18], строится модель на базе архитектуры VGG16 [19]. Часть слоёв импортируется с весами из подмодуля torchvision, далее средствами torch к ним добавляются новые слои. Обучение полученной модели также было осуществлено самостоятельно. Более подробно подготовка глубокой сети описана в деталях реализации.

С учётом заимствуемых фрагментов кода схема на рисунке 7 была расширена до диаграммы на рисунке 8. Все структурные блоки помимо красных были реализованы самостоятельно.

5.2 Исходные данные и условия эксперимента

В рамках эксперимента рассматриваются 2 модельные задачи miniImageNet и CIFAR100. В каждой задаче выделяется 80 классов, на которых происходит обучение генераторов по описанным алгоритмам. Оценка качества производится на 5 классах, не вошедших в тренировочные. В задаче CIFAR100 были взяты 5 связанных категорий, принадлежащих классу транспортных средств и передвижной техники, - газонокосилка, ракета, трамвай, танк, трактор. Для задачи miniImageNet рассматриваемые категории незави-

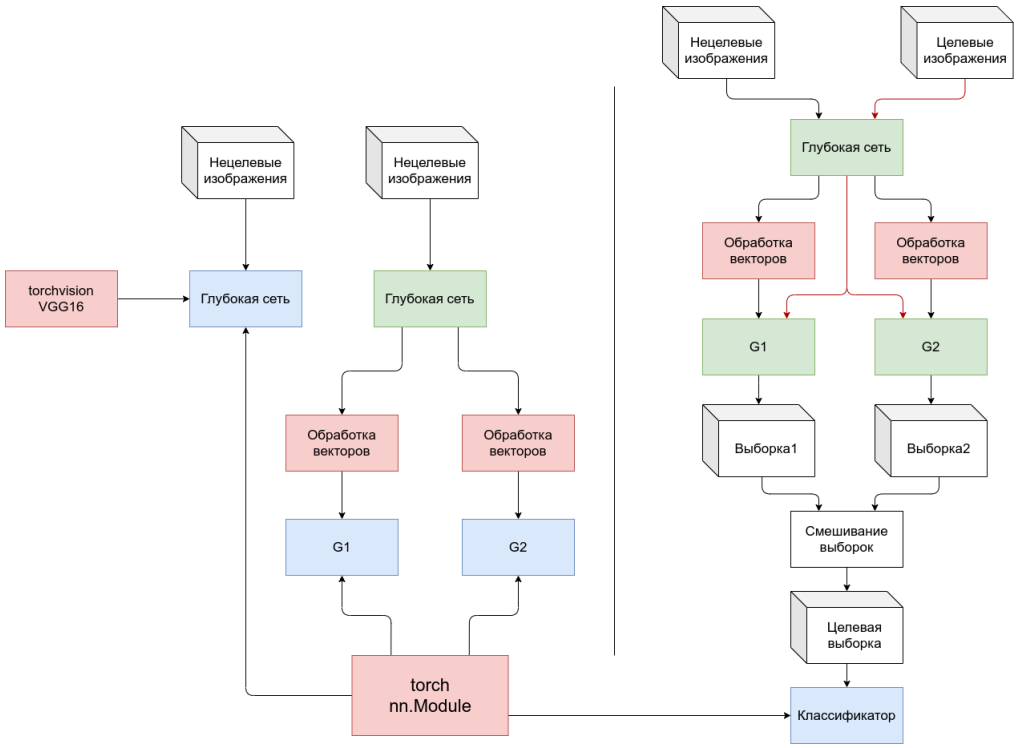

Рис. 8: Реализация решения. Синие блоки - обучение моделей, зелёные - использование, красные - заимствование кода/весов.

симы: автобус, электронное табло, сцена, ваза, торт. Изображения преобразуются в векторы пространства $\mathbb{R}^{2048}$. Каждым генератором воссоздаётся выборка из 1024 примеров, смешанная выборка также составляется из 1024 векторов. Целевая модель - многомерная логистическая регрессия. Метрика качества - точность $accuracy = correct/total$, где $total$ - количество объектов при валидации, $correct$ - количество верно классифицированных.

В ходе эксперимента выбирается некое соотношение для смешивания выборок. Для данного соотношения определённое количество раз k генерируется входная выборка. Входная выборка содержит по 1 или по 5 обучающих примеров на класс, сами классы в ходе экспериментов не меняются. Схема тестирования для некоторого соотношения выборок:

1. Фиксирование пропорции для смешивания.
2. Повтор k раз:
    3. Генерация входной выборки.
    4. Построение смешанной обучающей выборки.
        5. Обучение пакета классификаторов одной архитектуры.
        6. Усреднение точности классификации по пакету.
7. Усреднение точности по k генерациям входных выборок.

Такая схема позволяет получить среднюю оценку точности для некоторого соотношения при смешивании по различным целевым задачам. Во всех экспериментах $k = 32$. Указанная схема варьируется для всевозможных пропорций.

5.3 Детали реализации.

Для подготовки глубокой сети строится архитектура из предобученных свёрточных слоёв модели VGG16 [19], двух скрытых полносвязных слоёв размера 2048 с функцией активации $ReLU$ и некоторого выходного слоя. Полученная сеть обучается классификации изображений на нецелевых категориях выборки miniImageNet. Для генерации вектора признаков, с помощью обученной сети по изображению рассчитывается выход второго полносвязного слоя длины 2048.

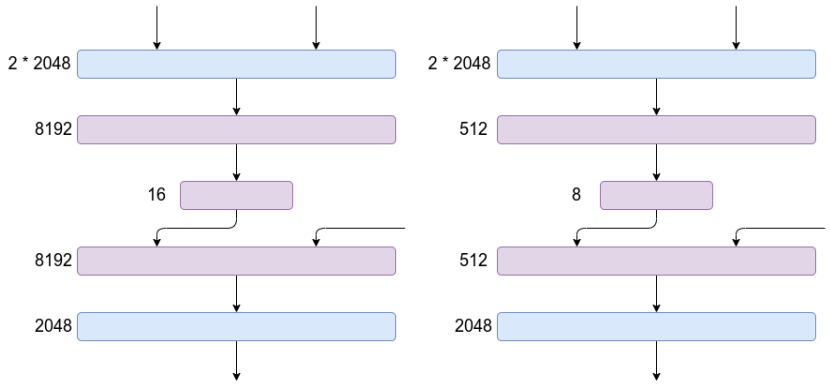

Рис. 9: Схемы реализации △-кодировщика для ***miniImageNet*** (слева) и для ***CIFAR100*** (справа). Рядом со слоями указано количество нейронов. Скрытые слои имеют фиолетовый цвет.

Каждая подсеть △-кодировщика представляет из себя 1 скрытый слой с функцией активации $lReLU = max(x, 0.2x)$. Схемы архитектур для miniImageNet и для CIFAR100 показаны на рисунке 9. Все слои реализации, кроме выходного, содержат случайное отключение нейронов с вероятностью 0.5. Функция ошибки $loss = weightedMAE(\widehat{X}, X) = \sum\limits_i w_i|\widehat{X}_i - X_i|$, где веса $w_i = |\widehat{X}_i - X_i|/||\widehat{X} - X||$ введены для борьбы с большими градиентами. Оптимизатор Adam со скоростью сходимости 10^{-5}.

Центроидный генератор составлен как полносвязная сеть с функциями активации $ReLU$. Схема архитектуры показана на рисунке 10. Функция ошибки $loss = MSE(\widehat{C}, C) = \frac{1}{N}\sum\limits_i^N(\widehat{C}_i - C_i)^2$. Оптимизация стохастическим градиентным спуском со скоростью сходимости 10^{-1}, моментом 0.9, $L2$ - регуляризацией с коэффициентом 0.0001.

14 эпох обучения глубокой сети длились около 3.5 часов, 10 эпох обучения △-кодировщика около 31с, 32 эпохи обучения центроидного генератора около 1мин 52с.

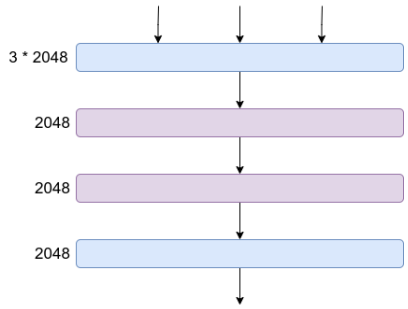

Рис. 10: Схема реализации центроидного генератора. Для обеих баз изображений архитектура одинаковая. Рядом со слоями указано количество нейронов. Скрытые слои имеют фиолетовый цвет.

Оптимизация проводилась по пакетам в 512 примеров для генераторов и в 100 примеров для глубокой сети.

Целевой классификатор реализуется как однослойный персептрон с функцией активации $softmax$. Функция ошибки - перекрёстная энтропия. Оптимизация методом Adam со скоростью сходимости 10^{-4}.

5.4 Результаты эксперимента

Для каждой базы изображений приведено 2 графика точности классификации при различных соотношениях смешиваемых выборок (рисунки 11 и 12). Левый и правый графики построены для случаев, когда для целевого класса дано, соответственно, по 1 и по 5 обучающих примеров. По оси x откладывается перевес числа примеров в пользу одного из генераторов при смешивании. Нуль означает смешивание выборок в равном соотношении, то есть 512 к 512 при итоговой выборке из 1024 примеров. Точки, отдаляющиеся от нуля, соответствуют всё большему преобладанию примеров одного из генераторов в выборке. Для краткости центроидный генератор обозначен как $G1$, $\triangle$-кодировщик как $G2$. Также для наглядности сравнения представлены рекордные точности для 3 типов выборок: воссозданной каждым из генераторов по отдельности (2 точности) и воссозданной ансамблем с усреднением по нескольким соотношениям в окрестности рекордного (таблицы 1 и 2).

Входных примеров	G1	G2	G1 + G2 (0.51 : 0.49)
1	58.9%	56.9%	64.2%
5	75.9%	74.5%	80.1%

Таблица 1: ***miniImageNet.*** Рекордная точность для 3 типов выборок. В скобках указана оптимальная пропорция.

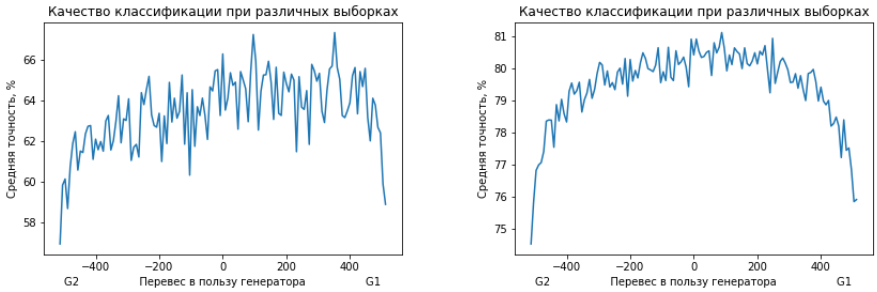

Рис. 11: ***miniImageNet.*** Точность для различных соотношений при смешивании.

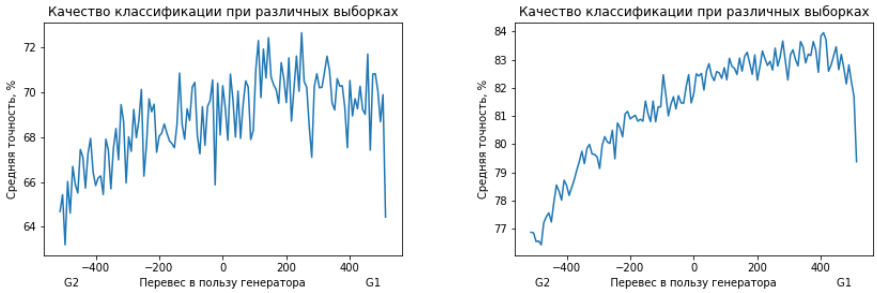

Рис. 12: ***CIFAR100.*** Точность для различных соотношений при смешивании.

Входных примеров	G1	G2	G1 + G2 (0.9 : 0.1)
1	64.4%	64.7%	69.2%
5	79.4%	76.9%	83%

Таблица 2: ***CIFAR100.*** Рекордная точность для 3 типов выборок. В скобках указана оптимальная пропорция.

5.5 Обсуждение и выводы

На основании эксперимента были сделаны выводы:

- Смешивание выборок, составленных различными генераторами, повышает качество решения на 3-6%. Прирост наблюдается и в случаях, когда имеется существенный перевес в пользу примеров одного из генераторов.

- Не всегда наилучшее качество достигается при смешивании синтетических выборок в равном соотношении. В приведённых экспериментах оптимальными соотношениями были 0.59:0.41 для независимых категорий miniImageNet и 0.9:0.1 для связанных классов CIFAR100.

- Данное соотношение является гиперпараметром предлагаемого ансамбля, так как выбор пропорции зависит от задачи.

- Во всех случаях решение задачи классификации ведёт себя неустойчиво при наличии 1 нового примера на класс.

6 Заключение

- В рамках работы был проведён обзор методов генерации искусственных выборок, решающих задачу классификации изображений по небольшому числу примеров.
- На основании обзора были выделены 2 эффективных метода, с помощью которых был построен ансамбль алгоритмов генерации.
- Данный ансамбль был реализован и протестирован при различных значениях возникающего гиперпараметра - пропорции смешивания.
- На основании тестирования были сделаны 2 основных вывода:
  1) смешивание выборок позволяет повысить качество классификации
  2) оптимальное соотношение при смешивании зависит от задачи.

7 Список литературы

1. Wei W. Yu Haonan Zhang H. Xu W. Wu Y. // MetaView: Few-shot Active Object Recognition. // Computing Research Repository, 2021, Abstract 2103.04242
2. Ayub A. Wagner A. R. // Tell me what this is: Few-Shot Incremental Object Learning by a Robot. // In Proceedings of the 33rd International Conference on on Intelligent Robots and Systems, Las Vegas, USA, 2020, P. 8344–8350
3. Ayub A. Wagner A. // F-SIOL-310: A Robotic Dataset and Benchmark for Few-Shot Incremental Object Learning. // Computing Research Repository, 2021, Abstract 2103.12242
4. Brown T. Mann B. Ryder N. Subbiah M. Kaplan J. D. Dhariwal P. Neelakantan A. Shyam P. Sastry G. Askell A. Agarwal S. Herbert-Voss A. Krueger G. Henighan T. Child R. Ramesh A. Ziegler D. Wu J. Winter C. Hesse C. Chen M. Sigler E. Litwin M. Gray S. Chess B. Clark J. Berner C. McCandlish S. Radford A. Sutskever I. Amodei D. // Language Models are Few-Shot Learners. // In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, Canada, 2020, P. 1877–1901

5. Russakovsky O. Deng J. Su H. Krause J. Satheesh S. Ma S. Huang Z. Karpathy A. Khosla A. Bernstein M. Berg A. C. Fei-Fei L. // ImageNet Large Scale Visual Recognition Challenge. // International Journal of Computer Vision, 2015, Volume 115, Issue 3, P. 211–252.

6. Oord A. Kalchbrenner N. Espeholt L. Kavukcuoglu K. Vinyals O. Graves A. // Conditional Image Generation with PixelCNN Decoders. // In Proceedings of the 29th International Conference on Neural Information Processing Systems, Barcelona, Spain, 2016, P. 4790–4798

7. Goodfellow I. Pouget-Abadie J. Mirza M. Xu B. Warde-Farley D. Ozair S. Courville A. Bengio Y. // Generative Adversarial Networks. // In Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, Canada, 2014, P. 3320–3328

8. Oord A. Dieleman S. Zen H. Simonyan K. Vinyals O. Graves A. Kalchbrenner N. Senior A. Kavukcuoglu K. // Wavenet: A Generative Model For Raw Audio. // Computing Research Repository, 2016, Abstract 1609.03499

9. Volpi R. Morerio P. Savarese S. Murino V. // Adversarial Feature Augmentation for Unsupervised Domain Adaptation. // In Proceedings of the International Conference of Computer Vision and Pattern Recognition, Salt Lake City, USA, 2018, P. 5495–5504

10. Frid-Adar M. Klang E. Amitai M. Goldberger J. Greenspan H. // Synthetic Data Augmentation Using GAN For Improved Liver Lesion Classification. // In Proceedings of the 15th International Symposium on Biomedical Imaging, Washington, DC, USA, 2018, P. 289-293

11. Wang Y. Yao Q. Kwok J. T. Ni L. M. // Generalizing from a Few Examples: A Survey on Few-Shot Learning. // ACM Computing Surveys, 2020, Volume 53, Issue 3, P. 1–34

12. Finn C. Abbeel P. Levine S. // Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. // In Proceedings of the 34th International Conference on Machine Learning, Stockholm, Sweden, 2017, P. 1126–1135

13. Vinyals O. Blundell C. Lillicrap T. Kavukcuoglu K. Wierstra D. // Matching Networks for One Shot Learning. // In Proceedings of the 30th International Conference on Neural Information Processing Systems, Barcelona, Spain, 2016, P. 3630–3638

14. Snell J. Swersky K. Zemel R. // Prototypical networks for few-shot classification. // In Proceedings of the 30th International Conference on Neural Information Processing Systems, Long Beach, USA, 2017, P. 4077–4087

15. Yosinski J. Clune J. Bengio Y. Lipson H. // How transferable are features in deep neural networks? // In Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, Canada, 2014, P. 3320–3328

16. Bae S.-H. Lee Y. Jo Y. Bae Y. Hwang J. // Rank of Experts: Detection Network Ensemble. // Computing Research Repository, 2017, Abstract 1712.00185

17. Schwartz E. Karlinsky L. Shtok J. Harary S. Marder M. Kumar A. Feris R. Giryes R Bronstein A. M // Delta-encoder: an effective sample synthesis method for few-shot object recognition. // In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montreal, Canada, 2018, P. 2850–2860.

18. Hariharan B. Girshick R. // Low-shot Visual Recognition by Shrinking and Hallucinating Features. // In Proceedings of the International Conference on Computer Vision, Venice, Italy, 2017, P. 3037–3046.

19. Simonyan K. Zisserman A. // Very Deep Convolutional Networks for Large-Scale Image Recognition. // Computing Research Repository, 2014, Abstract 1409.1556