Визуализация нейронных сетей и генерация изображений (часть 2)

Дмитрий Попов
Московский Государственный Университет им. М.В.Ломоносова
dr.dmitrii2000@yandex.ru

1 июня 2021 г.

Аннотация

Данная работа является второй частью конспекта лекции «Визуализация нейронных сетей и генерация изображений» по курсу «Глубокое обучение» и основана на материалах лекции, а также материалах, опубликованных в интернете по соответствующей теме. Рассмотрены основные идеи и методы визуализации работы нейронной сети при помощи генерации изображений. Рассмотрены задачи генерации изображений по признаковому описанию, генерации текстур, генерации пейзажей по сегментационной маске, задача переноса стиля.

I. Введение

Несмотря на сильный прогресс в обучении глубоких нейросетевых моделей, задача интерпретации принципов их работы не решена до конца. Детальное понимание того, как и почему работает модель, помогает видеть проблемы существующих и возможные направления развития новых подходов. Мощным инструментом, позволяющим человеку проинтерпретировать процесс работы всех этапов нейросети, является визуализация.

Идея генерации изображений при помощи нейросетевых моделей, появилась как средство визуализации работы нейросетей. Для визуализации выученного сетью класса, в работе [Simonyan et al., 2014] был предложен метод градиентного подъема в пространстве пикселей изображения для максимизации рейтинга класса. Подход получил развитие в [Nguyen et al., 2015], где были проведены исследования генерации изображений, максимизирующих активации отдельных нейронов, и предложены несколько способов регуляризации, для получения более интерпретируемых изображений. Эта регуляризация призвана решить проблему высокочастотного шума на сгенерированном изображении. Однако, из-за борьбы с высокочастотным шумом, страдают границы объектов - места где естественен резкий переход. Решение этой проблемы предложено в [Tyka, 2016], где была примененна идея использования фильтра специального рода – bilateral filter. Этот фильтр имеет свойство отфильтровывать высокочастотный шум, но сохранять края объектов. Другим методом улучшения качества генерации изображений, является переход при оптимизации в декоррелируемое пространство [Olah et al., 2017]. В [Mordvintsev et al., 2015], [Øygard, 2015] улучшено качество и интерпретируемость сгенерированных изображений, при помощи методов «transformation Robustness» (генерируемое изображения при небольших изменениях должно вызывать достаточно большие активации нейронов) и «learned Prior» (модель ориентируется на объекты нужного класса из обучения).

Задача генерация изображений получила развитие в ряде других направлений. Например, генерация изображений с заданным признаковым описанием [Dosovitskiy et al., 2017]. Этот метод позволяет визуализировать некоторый новый объект, признаковое описание которого может быть линейной комбинацией признаковых описаний нескольких других объектов.

Генерация изображений при помощи нейросетей имеет приложение в ряде таких задач, как генерация текстур [Gatys et al., Nov 2015], генерация пейзажа [Ulyanov, 2017] и перенос стиля [Gatys et al., Sep 2015]. В задаче генерации текстур требуется сохранить локальную структуру изображения, при том, что глобальная структура (например, расположение камней в кладке) может быть произвольной. Генерация пейзажей по сегментационной маске производится при помощи алгоритмов генерации текстур. В задаче переноса стиля требуется по изображению стиля определить параметризованное описание стиля и перенести его на требуемое изображение с сохранением его исходной структуры. Для решения данных задач

используются глубокие нейронные сети, обученные на классификацию изображений.

II. Основная часть

i. Визуализации нейросетей при помощи генерации изображений

Основная идея визуализации нейросетей при помощи генерации изображений заключается в поиске изображения, максимизирующего некоторые активации нейронной сети. Визуализацию можно проводить на разном уровне – от конкретных нейронов, до целых слоев. Различные цели оптимизации показывают, что ищут различные части сети. Возможные цели оптимизации и примеры сгенерированных изображений приведены на Рис. 1.

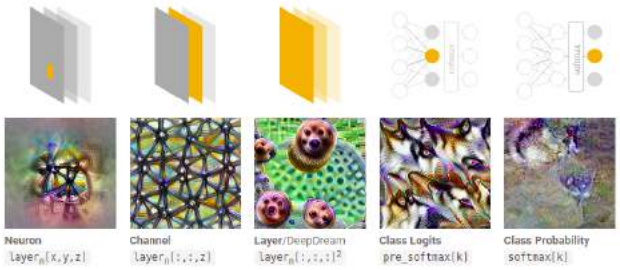

Рис. 1: Примеры генерации изображений для визуализации нейрона промежуточного слоя, канала промежуточного слоя, всего слоя, а также нейронов, отвечающих за рейтинг класса до и после применения softmax. На рисунке видна характерная особенность подобной визуализации: наиболее качественным получается изображение класса если генерировать изображений максимизирующее рейтинг класса до применения softmax. [Olah et al., 2017]

Подобный подход к визуализации работы нейросети позволяет понять, на что обращает внимание модель, и почему принимает то или иное решение о классификации. На Рис. 2 приведен пример визуализации нейронов и изображений из датасета на которых он максимально активируется [Olah et al., 2017]. На Рис. 2 видно, что один и тот же нейрон может соответствовать смеси нескольких идей. Эта проблема также продемонстрированна на Рис.3, где визуализация показывает неоднозначный результат. Нейрон одновременно активируется на картинках лис, котиков и машин. При этом четыре попытки визуализировать нейрон дают возможность для различной интерпретации его функций.

Мультимодальность нейронов исследована в [Goh, et al., 2021]. Мультимодальные нейроны реагируют на один и тот же объект на фотографиях, рисунках и изображениях с его именем. Т.е. нейрон «понимает» исследуемый образ. Кроме того, нейроны реагируют на темы, являющиеся

Рис. 2: Примеры генерации изображений для визуализации нейронов нейронной сети и изображения из датасета, которым отвечает максимальная активация этих нейронов. Первый нейрон одновременно улавливает полосы и мячи, второй – лица разных животных, похожих на собак, третий улавливает пушистую структуру облаков и каких то прочих объектов, четвертый – одновременно строения и небо. [Olah et al., 2017]

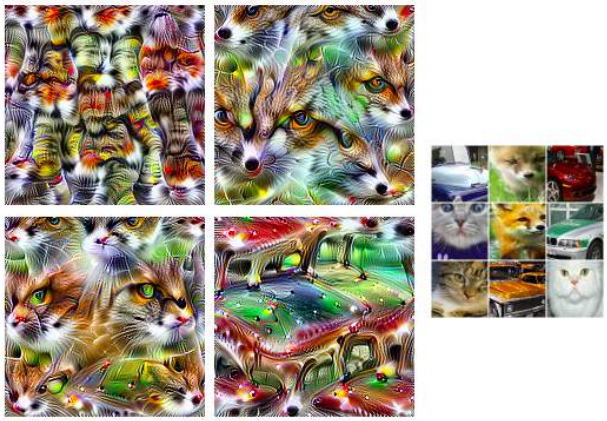

Рис. 3: На рисунке (слева) приведены 4 визуализации нейрона из некоторого слоя глубокой нейронной сети и (справа) изображения из датасета, которым отвечает максимальная активация этого нейрона. Нейрон активируется на изображениях кошек, лисиц и автомобилей, при этом каждая из 4х визуализаций в отдельности не дает возможности правильно проинтерпретировать функцию нейрона. [Olah et al., 2017]

контекстуально синонимичными рассматриваемому объекту (например: Е.И. Моисеев – функциональный анализ, БЭСМ-6). На Рис. 5 показано, как это проявляется. Интересный результат наблюдается при рассмотрении нейронов, отвечающих за эмоции. Поскольку небольшое изменение в выражении лица человека может радикально изменить смысл изображения, эмоциональное содержание имеет важное значение для задачи подписи. Модель посвящает этой задаче десятки нейронов, каждый из которых представляет различные эмоции. Эти эмоциональные нейроны не просто реагируют на выражения лица, связанные с эмоцией, - они также реагируют на язык тела, мимику, текст и рисунки Рис. 6.

Важным фактором в инерпретации работы нейросети является понимание того, как взаимодействуют различные нейроны, отвечающие за

Рис. 4: На рисунке приведены результаты совместной оптимизации двух нейронов. Получившийся результат генерации похож на интерполяцию между двумя изображениями, сгенерированными в результате оптимизации обоих нейронов по отдельности. [Olah et al., 2017]

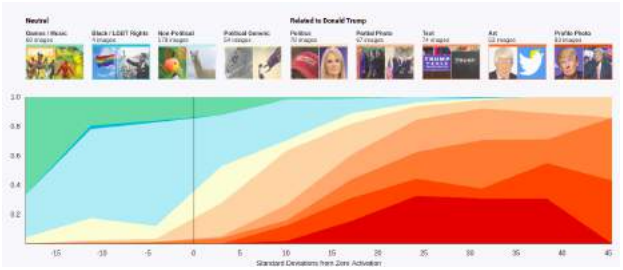

Рис. 5: Активации нейрона, отвечающего за Дональда Трампа на различных темах. Было собрано около 650 изображений, которые заставляют его срабатывать в разных количествах, и их вручную распределили по категориям. [Goh, et al., 2021]

различные признаки. В [Olah et al., 2017] приведен пример совместной оптимизации активаций двух нейронов Рис. 4.

Одним из способов интерпретации работы нейросети являются т.н. семантические словари [Olah, et al., 2018]. Семантический словарь можно построить для пространственных активаций нейронов любого слоя. Идея метода заключается в том, чтобы связать значение активации нейрона для некоторого входного изображения и визуализацию этого нейрона. Опишем процесс построения семантического словаря. Для некоторого входного изображения на слое l имеется $h_l \times w_l \times c_l$ активаций. h_l, w_l, c_l – высота, ширина и глубина соответственно. Фиксируя первые две размерности – i, j, получаем вектор пространственных активаций вида: $a^l_{i,j} = (a^{i,j}_1, ..., a^{i,j}_{c_l})$. Производится сортировка по величине активаций и выводятся пары – значение активации нейрона и его визуализация. Семантический словарь дает ответ на вопрос о том,

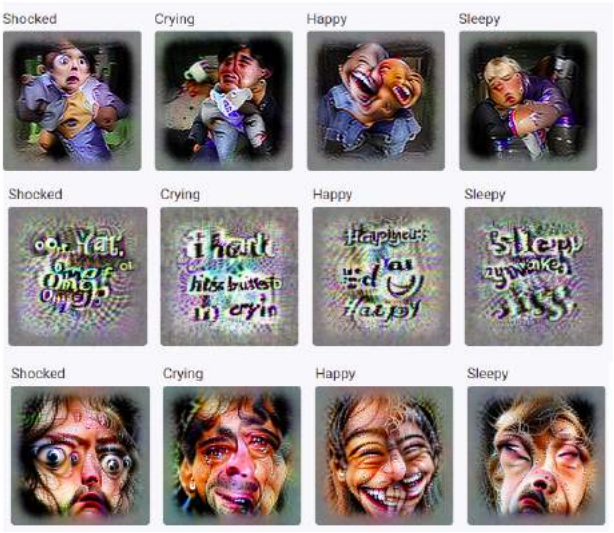

Рис. 6: Визуализации (сверху-вниз по строкам) различных эмоций по позе, тексту, лицу. [Goh, et al., 2021]

какие признаки нейросеть распознала в данной области изображения на каждом слое. Примеры семантических словарей приведены на Рис. 7.

Рис. 7: Слева приведено исходное изображение и рассматриваемая область. Справа – соответствующий некоторому скрытому слою сети и данной области семантический словарь в виде пар: визуализация нейрона, значение активации [Olah, et al., 2018]

Семантические словари дают нам детальный взгляд на активацию: что обнаруживает каждый отдельный нейрон? Основываясь на этом представлении, мы также можем рассмотреть вектор активации в целом. Вместо того, чтобы визуализировать отдельные нейроны, мы можем визуализировать комбинацию нейронов, которые срабатывают в заданном пространственном положении. Конкретно, мы оптимизируем изображение, чтобы максимизировать точечное произведение его активаций с исходным вектором активации. Пример приведен на Рис. 7. Если применить эту технику ко всем векторам активации, мы не только увидим, что сеть обнаруживает в каждой позиции, но и то, как сеть понимает входное изображение в целом. Мы можем наблюдать, как развивается понимание сети: от обнаружения ребер в более

ранних слоях до более сложных форм и частей объектов в последних. Пример показан на Рис. На Рис. 7.

Рис. 8: Слева приведено исходное изображение и рассматриваемая область. Справа – соответствующая семантическому словарю визуализация нейрона [Olah, et al., 2018]

Рис. 9: На рисунке показаны визуализации нейронов на нескольких слоях сети [Olah, et al., 2018]

Проблема этого подхода в том, что сеть показывает, как видит одно изображение. Но что, если мы хотим видеть более общую картину? Ответ на этот вопрос дается в [Carter, et al., 2019], где для понимания концепций взаимодействия нейронов предложено использование «атласа активаций». Идея заключается в использовании техники, аналогичной описанной ранее, но вместо отображения входных данных мы показываем визуализацию функций усредненных активаций. Объединив этот метод и метод снижения размерности, мы можем получить глобальную карту, которую «видит» сеть. Атласы активации дают широкий обзор карт признаков, отбирая большое количество образцов из множества вероятных активаций. В статье используется миллион изображений, выбранных случайным образом из обучающих данных набора данных ImageNet. Рассматривается сеть GoogLeNet. Опишем процедуру построения атласа активаций. Мы случайно выбираем одну пространственную активацию на изображение. Размерность этого вектора может достигать 512. При помощи алгоритмов снижения размерности (t-SNE или UMAP), мы проецируем эти вектора в двухмерное пространство. Далее мы рисуем сетку поверх созданного 2D–пространства. Для каждой ячейки этой сетки, мы усредняем все активации, которые находятся в пределах этой ячейки, и используем визуализируем каноническое представление. Алгоритм схематично описан на Рис. 10. Получающиеся атласы активаций имеют вид, аналогичный представленному на Рис. 11 Подробно можно рассмотреть атласы активации в [Carter, et al., 2019].

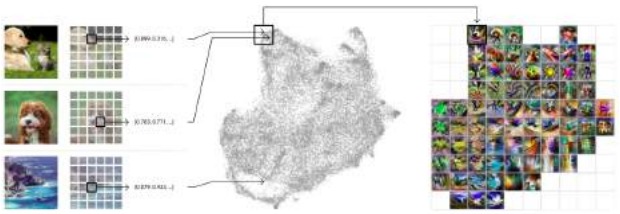

Рис. 10: (слева) Изображения передаются через сеть, собирая одну случайную пространственную активацию на изображение. (центр) Активации попускаются через UMAP, чтобы снизить размерность до двух. Эти точки строятся на графике. (справа) Затем мы рисуем сетку и усредняем активации, которые попадают в ячейку, и запускаем визуализацию признака на усредненной активации. Мы также дополнительно определяем размер ячеек сетки в соответствии с плотностью количества активаций [Carter, et al., 2019]

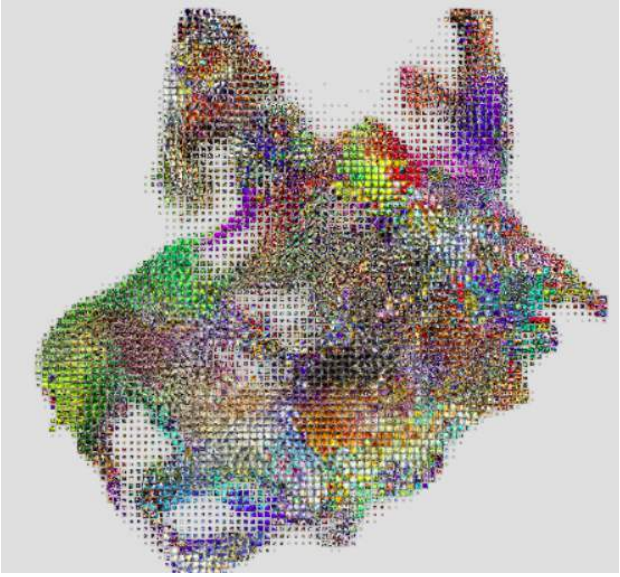

Рис. 11: Пример вида атласа активаций [Carter, et al., 2019]

История возникновения подходов к генерации изображений и их улучшений, а также сами эти подходы будут описаны в следующих разделах.

ii. Визуализация классов сети

В [Simonyan et al., 2014] был предложен алгоритм генерации изображения, максимально характеризующего некоторый класс нейросети. В работе предложено использование L_2–регуляризации для того, чтобы избежать накапливания больших входных значений в каких-то регионах изображения, которые максимизировали бы вероятность

требуемого класса.

Для заданного класса c, изображения I, рейтинг класса (до применения softmax) обозначается $S_c(I)$. При помощи градиентной оптимизации в пространстве пикселей входного изображения решается следующая оптимизационная задача:

$$argmax_I(S_c(I) - \lambda\|I\|_2^2)$$

Здесь λ–параметр регуляризации. Поскольку рассматриваемая сеть обучалась на 0-центрированных изображениях по датасету, в качестве начального приближения взято нулевое изображение. В статье предложено максимизировать рейтинг $S_c(I)$ класса c до применения softmax к рейтингам всех классов задачи, поскольку подход с оптимизацией вероятности $P_c(I)$ [1] класса c показал худший результат. На Рис. 12 приведены примеры визуализации нескольких классов.

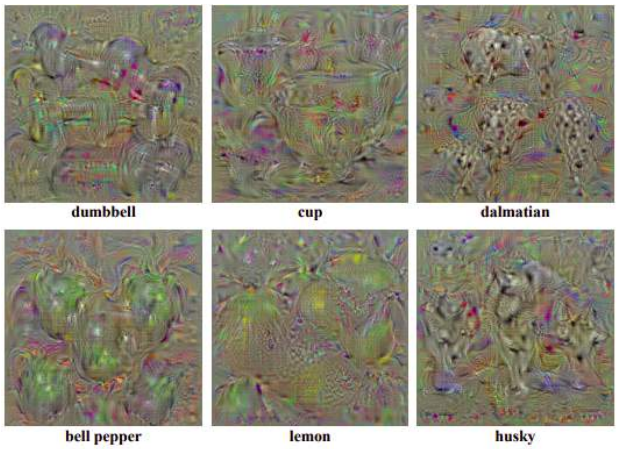

Рис. 12: Сгенерированные изображения, визуализирующие характерный внешний вид классов, изученных ConvNet, обученной на ILSVRC-2013. Заметно, как различные аспекты класса комбинируются в одном изображении. [Simonyan et al., 2014]

iii. Визуализация отдельных нейронов

Дальнейшее развитие направление генерации изображений для визуализации того, чему выучилась нейросеть, получило в работе [Nguyen et al., 2015]. В работе предложено несколько способов регуляризации, позволяющих получить более интерпретируемый результат, а так же проведены эксперименты по визуализации не только итоговых классов нейросети, но и промежуточных нейронов. Входное изображение обозначается x, значение активации некторого нейрона i для входа сети x обозначается $a_i(x)$, регуляризация обозначается $R_\theta(x)$, где θ-параметры регуляризации. Интересно, что регуляризация в этой статье реализована не по формуле $x^* = argmax_x(a_i(x) - R_\theta(x))$. Дополнительно вводится оператор регуляризации r_θ, применение которого чередуется с применением шага градиентного подъема. То есть формула шага градиентного подъема выглядит так:

$$x \leftarrow r_\theta(x + \eta\frac{\partial a_i}{\partial x})$$

Здесь η - шаг градиентного подъема. Кратко опишем способы регуляризации, предложенные в этой статье:

- L_2**–регуляризация** Эта регуляризация штрафует получающееся изображение за большие значения пикселей, и была применена еще в [Simonyan et al., 2014]. Оператор регуляризации имеет вид $r_\theta(x) = (1 - \theta_{decay}) \cdot x$. Параметр θ_{decay} характеризует силу L_2–регуляризации.
- **Размытие** Для борьбы с высокочастотным шумом на изображении, после нескольких шагов градиентного подъема применяется размытие Gaussian Blur. Оператор регуляризации имеет вид $r_\theta(x) = GaussianBlur(x, \theta_{blur\_width})$. θ_{blur_width} в данном случае - параметр, отвечающий за размер фильтра.
- **Обрезка малых по норме пикселей** Эта техника позволяет бороться с шумом на изображении, который не несет информации, однако значения пикселей малы по норме, и градиентный подъем не будет сильно изменять их значения. Оператор регуляризации r_θ в данном случае устанавливает в 0 значения всех пикселей, значения которых меньше некоторого заданного порога θ_{n_pct}.
- **Обрезка пикселей с малым вкладом** Эта техника помогает занулить несущественные пиксели. Вклад каждого пискеля изображения x оценивается как поэлементное произведение x на градиент $\nabla a_i(x)$. Оператор регуляризации r_θ в данном случае устанавливает в 0 значения всех пикселей, вклад которых меньше некоторого заданного порога θ_{c_pct}.

В статье указано, что каждый из типов регуляризации помогает бороться со своей патологией в генерации изображений, однако наиболее интерпретируемый результат получается для комбинации регуляризаций разных типов. Примеры получившихся изображений на некоторых слоях приведены на Рис. 13.

В [Nguyen et al., 2015] в качестве одного из методов визуализации работы нейронной сети предложен способ наблюдения изменений карт активаций

[1] $$P_c(I) = \frac{exp(S_c(I))}{\sum_i exp(S_i(I))}$$

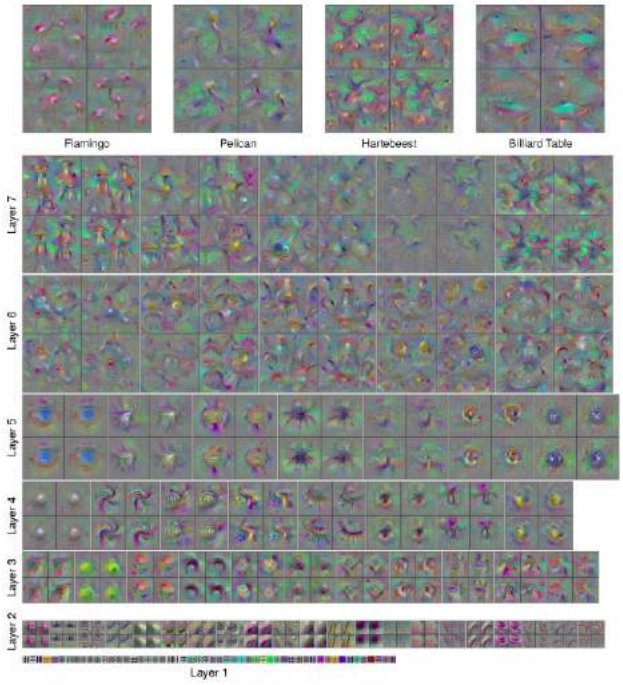

Рис. 13: Визузализация примеров признаков из восьми слоев глубокой сверточной нейронной сети. Изображения отражают истинные размеры объектов на разных слоях. В каждом слое представлены визуализации из 4 случайных запусков градиентной оптимизации для каждого канала. Визуализации показывают увеличение сложности и вариативности на более высоких слоях, состоящих из более простых компонентов из более низких слоев. [Nguyen et al., 2015]

промежуточных слоев нейронной сети для видео/последовательности изображений, пропущенных через сеть. Подобная визуализация помогает объяснить работу нейросети. При помощи анализа изменений карт активации нейронов одного из скрытых слоев был обнаружен нейрон Рис.14, который детектирует лица людей и зверей, однако у сети, обученной на классификацию не было отдельного класса «лицо». Таким образом, сеть посчитала полезной формирование признака «лицо» для дальнейшей классификации.

iv. Значительное увеличение качества генерируемых изображений

В [Mordvintsev et al., 2015] показано, как метод «transformation robustness» (устойчивость к преобразованиям) помогает повысить интерпретируемость генерации изображения при визуализации классов нейронной сети. Идея метода заключается в том, чтобы генерировать такие изображения, которые будут вызывать высокие значения активации даже при незначительном изменении (например, сдвиге на несколько пикселей). Подобный прием также называют «jitter» (дрожанием). Еще один важный прием примененный в работе – «learned priors». Это попытка модели изучить и

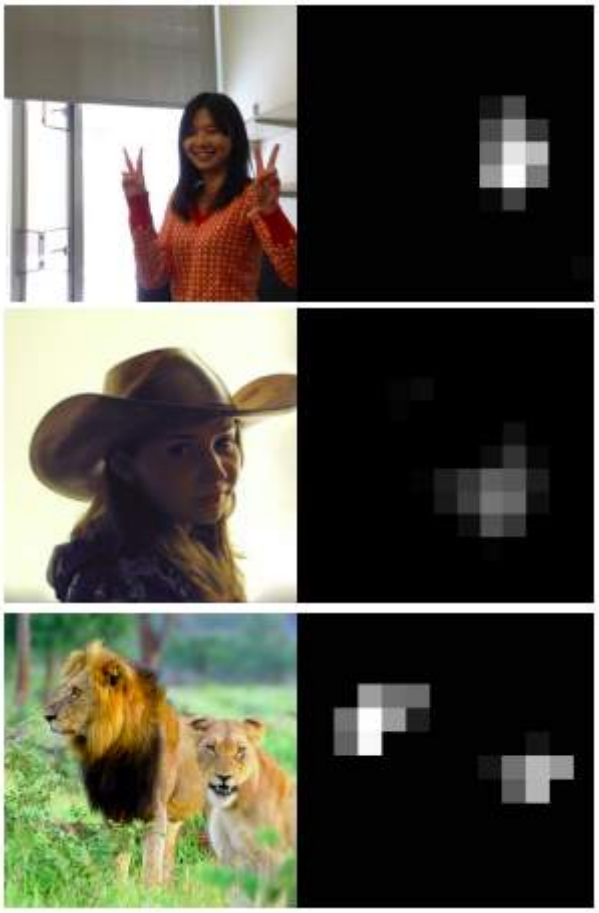

Рис. 14: Представление активаций внутреннего канала на скрытом сверточном слое глубокой нейронной сети, обученной на ImageNet. Набор данных не содержит класс лиц, но содержит много изображений с лицами. Канал реагирует на лица людей и животных и устойчив к изменениям масштаба, позы, освещения и контекста. [Nguyen et al., 2015]

сохранить при генерации структуру реальных изображений. Для этого используется совместная оптимизация генерации изображения и априорного распределения пикселей на реальных изображении. Пример генерации изображения для класса «банан» при помощи использования корреллированности соседних пикселей, свойственных для реальных изображений показан на Рис. 15.

Больше примеров визуализации классов приведено на Рис. 16.
В этой работе показан пример того, как визуализация помогает увидеть недостатки нейросети. Один из таких недостатков продемонстрирован на Рис. 17 .

Развите подход с комбинацией методов «jitter» и регуляризации выскочастотного шума получил в работе [Øygard, 2015]. При генерации изображения, в работе предложено сначала оптимизировать низ-

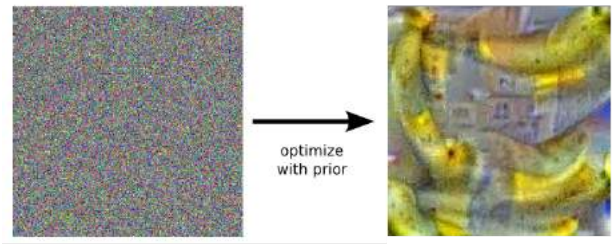

Рис. 15: Визуализация класса сети «Банан». Оптимизация начиналась со случайного шума, и проводилась с использованием требования о том что соседние пиксели должны быть скорреллировны с реальными изображениями из датасета. [Mordvintsev et al., 2015]

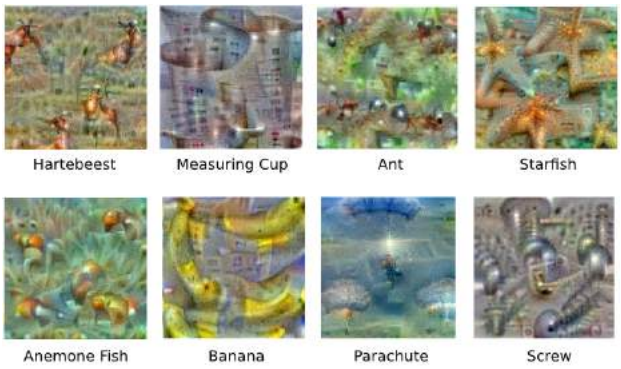

Рис. 16: Несколько примеров визуализации разных классов при помощи генерации изображений с описанными в статье приемами для улучшения интерпретируеммости. [Mordvintsev et al., 2015]

кочастотную информацию, которая даст общую структуру изображения, а затем постепенно вводить высокочастотные детали по мере продолжения градиентного подъема, фактически «размывая» изображение. При медленном применении этой техники, удается добиться хорошего качества оптимизации изображения Рис. 18. В работе применены два ключевых подхода:

- Применение гауссовского размытия после шагов градиентной оптимизации. При этом размер ядра размытия начинается с большого значения и медленно убывает.
- (нововведение) Применение гауссовского размытия к градиенту, начиная с большого размера ядра и медленно уменьшая его по мере итерации.

Дополнительно используется L_2–регуляризация пикселей, для постепенного уменьшения нерелевантного шума от предыдущих шагов оптимизации.

Отметим, что в этой работе не используются данные о распределении пикселей на реальных изображениях, и все сгенерированное изображение поступает исключительно из нейросети. Напротив, при генерации изображения при помощи техники «Learned priors» возникает трудность с пониманием

Рис. 17: Несколько примеров визуализации класса «Гантели». Видно, что все изображения содержат руку, держащую гантелю. Таким образом сеть запомнила объект, который часто встречается рядом с гантелей, но, вообще говоря, не характеризует класс. Подобный недостаток может привести к ошибкам классификации. [Mordvintsev et al., 2015]

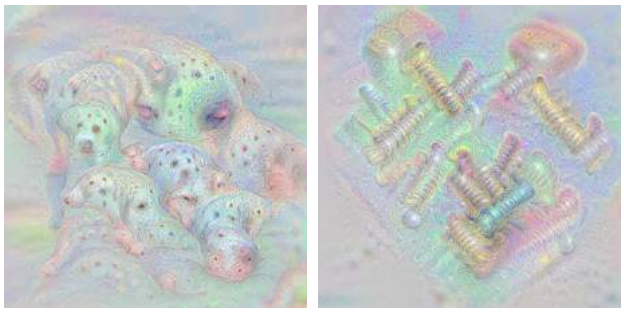

Рис. 18: Визуализация классов «Далматинец» (слева) и «Винты»(справа) [Øygard, 2015]. Видно, что далматинцы в основном определяются точками, нейросеть посчитала остальные признаки несущественными для классификации. Винты же, как более простой объект представлен полностью на сгенерированном изображении класса.

того, какая информация на изображении пришла из нейросети, а какая – из датасета.

В статье есть пример Рис. 19, где визуализация класса помогла определить недостаток в обучении нейросети.

Рис. 19: Визуализация класса «Саксофон», [Øygard, 2015]. Видно, что на сгенерированном изображении присутствует человек, держащий инструмент. Вероятно, это связано с тем, что в большинстве примеров изображений, используемых для обучения, был саксофонист, однако характерным объектом класса он не является. Такой недостаток также может привести к ошибке при классификации.

v. Inceptionism & Deep dreams

В [Mordvintsev et al., 2015] предложен оригинальный способ для понимания того, какие признаки на

изображении обрабатываются в том или ином слое. Метод заключается в том, чтобы для некоторого фиксированного слоя нейронной сети и фиксированно входного изображения при помощи градиентного подъема увеличить максимальные активации нейронов в слое. Таким образом, мы даем сети изображение и хотим увидеть, на что она обращает внимание в том или ином слое. Например, нижние слои, как правило, отвечают за штрихи или простые орнаментальные узоры, поскольку эти слои чувствительны к краям и ориентации объектов. Пример приведен на Рис. 20.

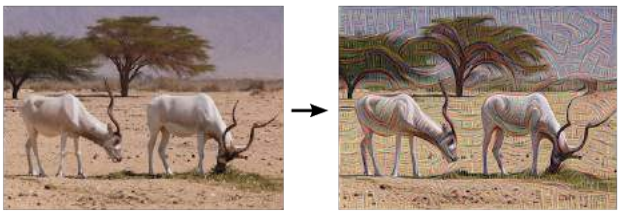

Рис. 20: Пример изменения изображения для усиления максимальных активаций нижних слоев глубокой сверточной сети из [Mordvintsev et al., 2015]. На данном уровне абстракции сеть обращает внимание на простые геометрические формы и штрихи.

Более высокие слои, напротив, как правило обращают внимание на более сложные сущности. Например это могут быть целые объекты. На Рис. 21 приведен пример применения подобного алгоритма для высокого слоя нейросети.

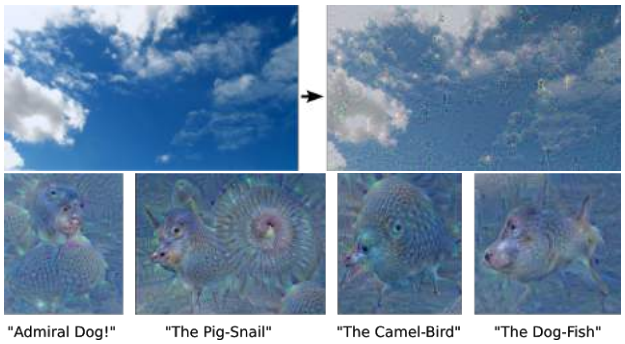

Рис. 21: Пример изменения изображения для усиления максимальных активаций высоких слоев глубокой сверточной сети из [Mordvintsev et al., 2015]. Сеть «замечает» в облаках объекты, которые «знает», и преобразует изображение так, чтобы эта схожесть была более заметна. На данном примере сеть деформировала часть облаков так, чтобы они стали похожи на абстрактных животных.

Подобный принцип назван авторами «Inceptionism». Предполагается, что он позволяет понимать уровень абстракции, которую достигла сеть. Интересный пример применения этой техники продемонстрирован на Рис. 22

Предположим, что мы начинаем применять данный алгоритм итеративно, начиная со случайного шума. При добавлении некоторого масштабирования после каждой итерации, мы получим последовательность изображений, созданных сетью, т.е набор признаков сети, которые сеть распознала среди случайного шума. Результат подобной генерации основан только на работе нейронной сети. Подобный алгоритм получил название «deep dreams», примеры его применения продемонстрированы на Рис. 23.

Рис. 22: Пример преобразования изображения с помощью принципа «Inceptionism» [Mordvintsev et al., 2015]. Сеть «увидела» в листьях силуэты птиц, в бурлящей реке и камнях – шерсть и глаза собак.

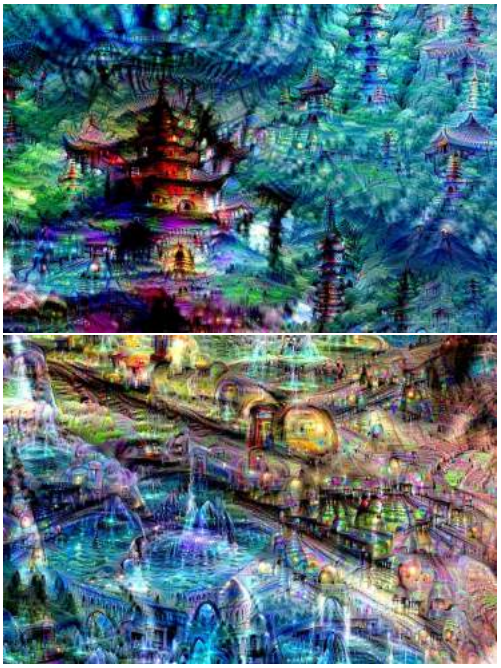

Рис. 23: Примеры генерации изображений из случайного шума с помощью алгоритма «deep dreams» [Mordvintsev et al., 2015]

vi. Декореллированное пространство

В предыдущих работах предложены идеи, ограничивающие высокочастотный шум на генерируемом

изображении, однако при они могут формироваться, когда градиент постоянно «подталкивает» их. Одним из способов изменить локальный максимум, в который попадет градиентная оптимизация, для того, чтобы избавиться от высокочастотного шума, является оптимизация в декореллированном пространстве. Подобная идея описана в [Olah et al., 2017] и названа «preconditioner» (предобуславливанием) градиента. Т.е. хотелось бы сделать данные декорелированными и не зашумленными. Предлагается проводить градиентный спуск в базисе Фурье с одинаково масштабированными частотами. Отметим, что необходимо обеспечить декореллированность между цветами. Для этого, явно измеряем корреляцию между цветами в обучающем наборе и используют разложение Халецкого для их декорреляции. Продемонстрируем на Рис. 24, как переход в декореллированное пространство помогает улучшить качество изображения.

Рис. 24: Пример того, как использование декорреллированного направления градиентоного спуска приводит к более хорошему результату [Olah et al., 2017]

vii. Bilateral filter

Bilateral filter [Tyka, 2016] – это простая схема сглаживания изображения с сохранением границ. При гауссовском размытии, в качестве нового значения пикселя, мы берем средневзвешенное значение значений пикселей в окрестности, где веса обратно пропорциональны расстоянию от центра окрестности. Помимо этих пространственных весов, bilateral filter добавляет новый «тональный» вес, такой, что веса пикселей, который близки по значению к пикселю в центре, больше, чем веса пикселей, значения которых сильно отличаются от значения пикселя в центре. Пример получаемого изображения приведен на Рис. 25

viii. Использование генеративной модели для реалистичной визуализации

В [Nguyen et al., 2016] при помощи техники «learned prior», значительно улучшено качество

Рис. 25: Синтез изображения класса «Пеликан» с использованием bilateral filter и сети GoogLeNet [Tyka, 2016]

генерации изображений. Алгоритм генерирует изображения, которые выглядят почти реальными Рис. 26, раскрывает особенности, изученные каждым нейроном, интерпретируемым способом Рис. 28. Для генерации реалистичных изображений необходимо ограничить область оптимизации, чтобы полученное синтетическое изображение одновременно максимизировало активации и было интепретируемым.

В статье предложено вместо ручного проектирования априорного распределения пикселей (как это было в [Mordvintsev et al., 2015]) использование улучшенного метода «learned prior», похожего на генеративную модель изображений. Генератор изображений – глубокая нейронная сеть, обученная принимать код (например, скалярный вектор) и выводить синтетическое изображение, которое выглядит как можно ближе к реальным изображениям из набора данных ImageNet. Таким образом, оптимизируя не непосредственно пиксели изображения, а входной код генеративной модели, мы получаем реалистичное изображение, которое максимизирует искомую активацию нейрона. Архитектура модели схематично описана на Рис. 27

ix. Генерация изображений: восстановление из признаков

В [Dosovitskiy et al., 2017] предлагается использование глубокой сверточной сети для генерации изображения по признаковому описанию. Задача ставится следующим образом: необходимо с помощью нейронной сети генерировать точные изображения объектов из высокоуровневого описания: стиль, ориентация по отношению к камере и прочие парамет-

Рис. 26: Изображения, синтезированные с нуля, сильно активирующие выходные нейроны в глубокой нейронной сети CaffeNet, обученной на классификацию изображений из датасета ImageNet. [Nguyen et al., 2016]

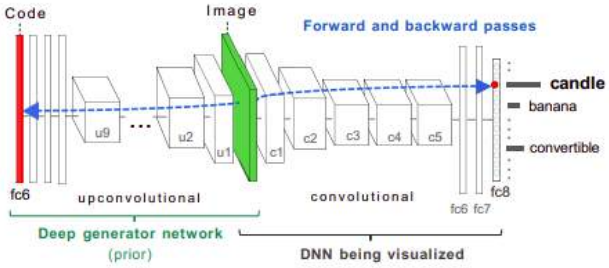

Рис. 27: Чтобы синтезировать предпочтительный вход для некоторого целевого нейрона h (например, выходной нейрон класса “свеча”), мы оптимизируем вход скрытого кода (красная полоса) нейросети – генератора изображений (DGN) для получения изображения, которое сильно активирует h. В приведенном примере DGN – это сеть, обученная инвертировать представления объектов слоя fc6 CaffeNet. Нейросеть DNN, которую мы хотим визуализировать может быть другой сетью (с другой архитектурой и или обученной на других данных). Информация о градиенте (синяя пунктирная линия) течет от слоя, содержащего h в целевом DNN (здесь слой fc8), через изображение обратно к входному кодовому слою DGN. DNN и DGN имеют фиксированные параметры, и оптимизация изменяет только входной код DGN (красная полоса). [Nguyen et al., 2016]

Рис. 28: Визуализация примеров детекторов нейронных признаков из всех восьми слоев глубокой нейронной сети CaffeNet. Изображения отражают истинные размеры восприимчивых полей на разных уровнях. Для каждого нейрона показаны 4 различные визуализации: верхние 2 изображения взяты из предыдущих работ авторов; а нижние 2 изображения взяты из описанного метода. Это параллельное сравнение показывает, что оба метода часто согласуются с особенностями, которые нейрон научился обнаруживать. В целом, описанный метод дает более реалистичный цвет и текстуру. Однако сравнение также говорит о том, что описанный метод плохо визуализирует лица животных (3-й и 4-й блок на слое 6; 1-й блок на слое 5; и 6-й блок на слое 4). [Nguyen et al., 2016]

ры, такие как цвет, яркость и т.д. На вход сети идут кортежи из трех векторов: c – определение стиля, v – положение камеры относительно объекта и θ – дополнительные параметры. Искусственные трансформации T_θ , задаваемые вектором θ, добавляются для увеличения объема обучающей выборки и для борьбы с переобучением, аналогично аугментациям данных при обучении сверточных нейронных сетей. Каждая трансформация T_θ является комбинацией следующих: вращение в плоскости (до $\pm 12 \circ$), сдвиг (до $\pm 10\%$ от размера изображения), увеличение (100% to 135%), растяжение по горизонтали или вертикали (до 10%), изменение оттенка (произвольный случайный аддитивный фактор), изменение насыщенности (от 25% до 400%), изменение яркости (от 35% до 300%). Сеть обучалась на 3D моделях столов, стульев и машин. Ее архитектура представлена на Рис. 29. С помощью данной глубокой сети строится пара: целевое изображение и его сегментационная маска. Эту сеть можно использовать для создания новых объектов путем смешивания нескольких объектов из обучающего набора. На Рис. 30 продемонстрирована работа построенной модели и показано влияние параметра θ. На Рис. 31 показано, что нейросеть генерирует ожидаемые изображения объектов при интерполяции входных параметров нескольких существующих моделей. То есть в построенном признаковом пространстве существует арифметика.

x. Генерация текстур

Одно из направлений, где оказались применимы нейросети обученные на классификацию изображений это генерация текстур [Gatys et al., Nov 2015]. Цель синтеза текстур состоит в том, чтобы вывести процесс генерации из примера текстуры, который затем позволяет создавать произвольное количество новых образцов этой текстуры. Существует два основных подхода к процесса генерации текстур. Первый подход заключается в создании новой текстуры путем пересчета либо пикселей, либо целых участков исходной текстуры. Однако, подобные методы не определяют фактиче-

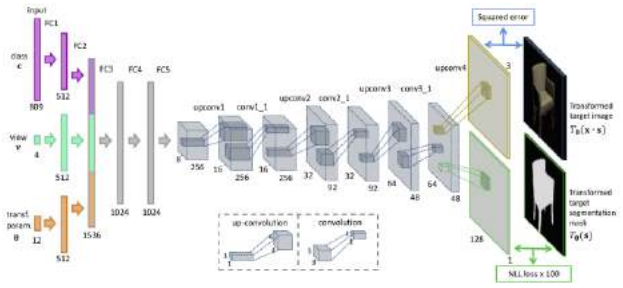

Рис. 29: Архитектура 1-потоковой глубокой сети («1s-S-deep»), которая генерирует изображения размером 128 × 128 пикселей. [Dosovitskiy et al., 2017]

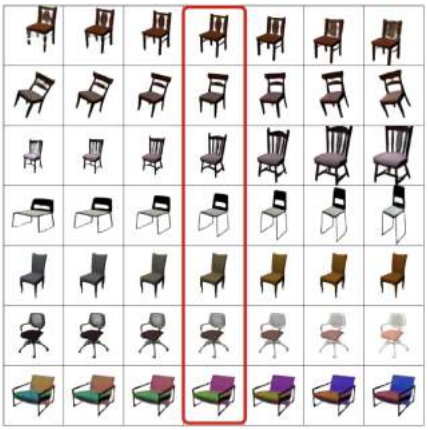

Рис. 30: Генерация изображений стула при активации различных преобразований. Каждая строка показывает одно преобразование: сдвиг, поворот, масштабирование, растяжение, насыщенность, яркость, цвет. В средней колонке показана реконструкция без каких-либо преобразований. [Dosovitskiy et al., 2017]

скую структуру текстуры, а скорее рандомизированно ее семплируют. Другой подход заключается в явном определении параметрической модели текстуры. В [Gatys et al., Nov 2015] предложена параметрическая текстурная модель с использованием сверточной нейронной сети для извлечения структуры текстурной модели. В качестве сверточной нейронной сети в работе использовалась VGG-19, обученная для распознавания объектов. Однако последние полносвязные слои не использовались, и max-pooling был заменен на average-pooling, который показал лучшие результаты в экспериментах. В силу того, что в НС используются только сверточные слои – размер входного изображения может быть любым. Архитектура получившейся НС и процесс синтеза текстуры схематично продемонстрированы на Рис. 32.

Опишем процесс построения матриц Грамма для входной текстуры x. Вектор x пропускается через сверточную нейронную сеть и вычисляются активации для каждого слоя l в сети. Пусть на выходе слой l выдает изображение с N_l различными каналами, каждый из которых после вытягивания матрицы в вектор имеет размер M_l. Тогда, эти признаки могут храниться в матрице $F^l = \mathcal{R}^{N_l \times M_l}$, $F_{i,j}$ i-ый элемент j-го канала тензора со слоя l. Описание текстуры должно опираться только на локальные особенности изображения, поэтому его можно задать как корреляции между различными признаками слоя. Эти корреляции с точностью до множителей задаются матрицей Грама, элементы которой определяются по формуле:

$$G^l_{i,j} = \sum_k F^l_{i,k} F^l_{j,k}$$

Набор матриц Грамма $G^1, ..., G^L$ со слоев $1, ..., L$, полученных в сети для заданной текстуры x, задает стационарное описание текстуры в данной модели.

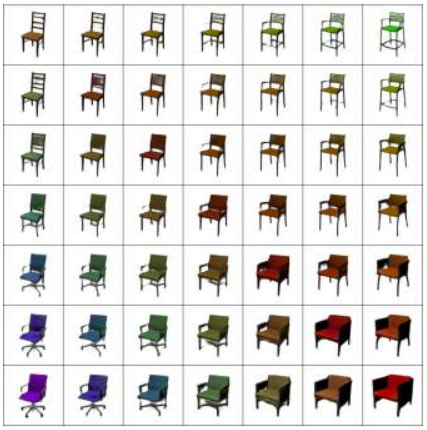

Рис. 31: Интерполяция между различными изображениями стульев. Модели, находящиеся в углах присутствуют в обучающем наборе, все остальные изображения «придуманы» сетью. [Dosovitskiy et al., 2017]

Для генерации изображения с нужной текстурой необходимо найти значения его пикселей. Берется изображение случайного шума $\hat{x}$. Пусть G_l и $\hat{G}_l$ матрицы Грама для слоя l для исходного и сгенерированного изображения соответственно. Тогда вклад слоя l в общую функцию потерь имеет вид:

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G^l_{ij} - \hat{G}^l_{ij})^2$$

Общая функция потерь принимает вид:

$$E^L = \mathcal{L}(x, \hat{x}) = \sum_{l=0}^{L} w_l \cdot E_l,$$

где w_l – весовые коэффициенты вклада каждого слоя в общую потерю. Производная E_l по активациям слоя l может быть вычислена аналитически:

$$\frac{\partial E_l}{\partial \hat{F}^l_{ij}} = \begin{cases} \frac{1}{N_l^2 M_l^2} \left((\hat{F}^l)^T (G^l - \hat{G}^l) \right)_{ij}, & \hat{F}^l_{ij} > 0 \\ 0, & \hat{F}^l_{ij} < 0 \end{cases}$$

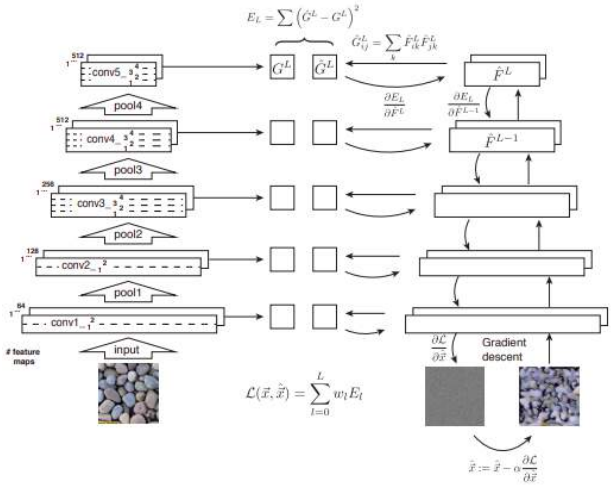

Рис. 32: На рисунке продемонстрирован метод работы алгоритма. Анализ текстуры (слева). Исходная текстура передается через CNN, и по признакам с нескольких слоев вычисляются матрицы Грамма G^l. Синтез текстур (справа). Изображение белого шума передается через CNN, и функция потерь E_l, равная L_2–норме разности матриц Грамма G^l исходной текстуры и $\hat{G}^l$ для генерируемого изображения. Функция общих потерь E_L представляет собой взвешенную сумму вкладов E_l от каждого слоя. Используя градиентный спуск для E_L по отношению к значениям пикселей, строится новое изображение, которое снова подается на вход сети. Таким образом итеративно генерируется текстура. [Gatys et al., Nov 2015]

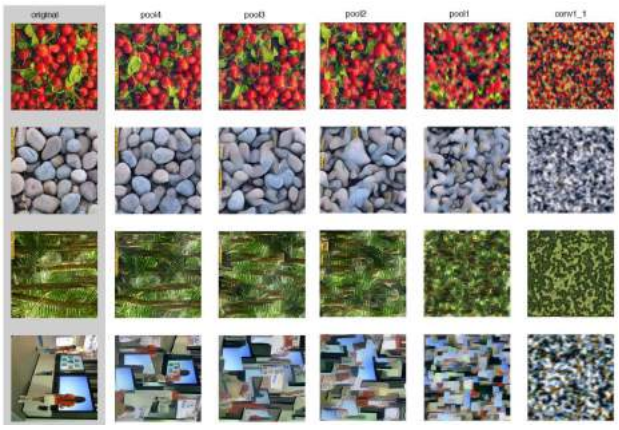

Рис. 33: Каждый столбец соответствует выбору слоев для описания текстуры. При использовании только первых слоев картинки имеют зернистую структуру, похожую только цветами. С увеличением числа слоев, на которых мы сопоставляем представление текстуры, мы обнаруживаем, что генерируем изображения с возрастающей степенью естественности. В последней строке показаны текстуры, созданные из нетекстурного изображения, чтобы лучше понять, как текстурная модель представляет информацию об изображении. [Gatys et al., Nov 2015]

Рис. 34: Влияние числа параметров модели на качество генерации текстуры [Gatys et al., Nov 2015]

Используя это, при помощи метода обратного распространения ошибки вычисляется производная $\mathcal{L}(x, \hat{x})$ по пикселям изображения $\hat{x}$. С помощью градиентного спуска находятся значения пикселей для искомой картинки.

На Рис. 33 показано влияние выбора слоев для описания текстуры. При использовании только первых слоев картинки имеют зернистую структуру, похожую только цветами. Но чем больше слоев, используется слоев, тем качественнее становится картинка. Последняя строка использует изображение, на котором нет текстуры, чтобы показать как нейросеть показано влияние числа параметров модели. Чем их больше, тем подробнее и реалистичнее изображение. Рис. 34.

xi. Генерация пейзажей

Одной из областей применения задачи генерации текстур, является генерация пейзажей. По заданной сегментационной маске, модель генерирует различные текстуры в разные области, создавая маски полноценный пейзаж. Пример работы такой модели взят из проекта [Ulyanov, 2017]. и изображен на Рис. 35

xii. Перенос стиля

Задача стилизации заключается в генерации изображения со стилем одного изображения и содержанием другого. В [Gatys et al., Sep 2015] рассмотрен способ использования нейросети, обученой на распознавание объектов для решения задачи переноса стиля.

Информация о входном изображении, содержащуюся в каждом слое глубокой сверточной нейронной сети позволяет реконструировать изображение по картам признаков. Более высокие уровни в сети захватывают высокоуровневый контент с точки зрения объектов и их расположения на исходном изображении, но не ограничивают точные значения пикселей. Признаки нижних слоев напроив, воспроизводят точные значения пикселей исходного изображения. Поэтому, признаки извлеченные из изображения в высоких слоях сети можно считать признаковым описанием контента.

Для получения представления о стиле входного изображения, мы используем признаковое пространство нескольких первых слоев, сохраняющее локальную информацию, аналогично подходу

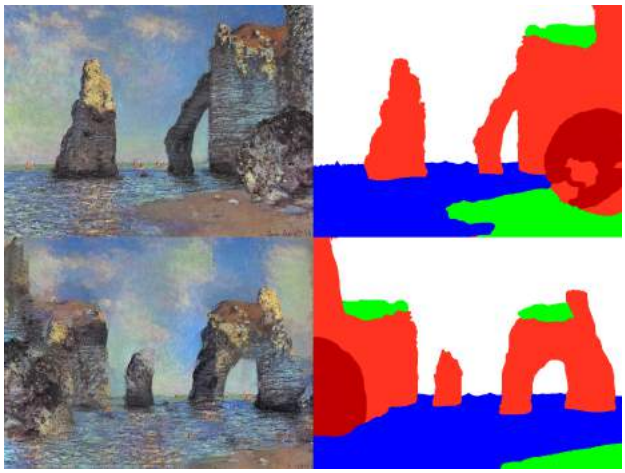

Рис. 35: Пример генерации пейзажей из [Ulyanov, 2017]. Слева сгенерированное изображение, справа – сегментационная маска.

у генерации текстур. Признаковым описанием стиля будет набор матриц Грама на выделенных слоях нейросети. Как было показано ранее в задаче генерации текстур, лучше брать комбинацию нескольких слоев для более детального описания стиля. Основным предположением является то, что представление содержания и стиля в сверточной нейронной сети являются разделимыми. Этот факт используется для создания новых изображений с нужным стилем и содержанием. Концепция работы алгоритма схематично описана на Рис. 36.

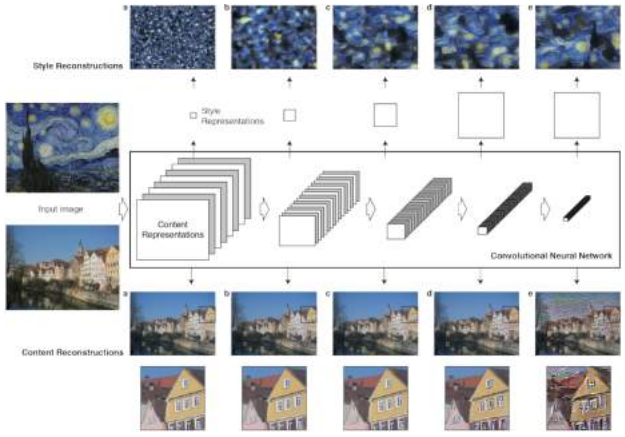

Рис. 36: Входное изображение представляется как набор признаков на каждом этапе работы сверточной сети. Пока число фильтров растет в процессе обработки изображения, размер изображения уменьшается. Чтобы провести реконструкцию контента, применяется реконструкция по верхним сверточным слоям VGG-19, которые сохраняют информацию о контенте на высоком уровне абстракции. Для реконструкции стиля вычисляется разница между признаками, отвечающими за локальную структуру генерируемого изображения и изображения стиля (т.е. матрица Грамма). Этот подход создает изображения, которые соответствуют стилю данного изображения в увеличивающемся масштабе, отбрасывая при этом информацию о глобальном расположении сцены. [Gatys et al., Sep 2015]

Пусть $\vec{p}$ и $\vec{x}$ исходное по содержанию изображение и сгенерированное изображение соответственно, а P^l и F^l – их признаковые описания нейросети на слое l соответственно. Определим квадратичную функцию ошибки между двумя представлениями. Это будет функция потерь по содержанию (content) на слое l.

$$\mathcal{L}_{content}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} (F^l_{i,j} - P^l_{i,j})^2$$

Производная этой функции потерь по активациям в слое l, имеет вид:

$$\frac{\partial \mathcal{L}_{content}}{\partial F^l_{i,j}} = \begin{cases} (F^l - P^l)_{i,j}, & F^l_{i,j} > 0 \\ 0, & F^l_{i,j} < 0 \end{cases}$$

Исходя из этого выражения, градиент по отношению к изображению $\vec{x}$ может быть вычислен с использованием обратного распространения ошибки. Таким образом, мы можем изменять исходное случайное изображение $\vec{x}$ до тех пор, пока оно не вызовет тот же отклик в определенном слое CNN, что и исходное изображение $\vec{p}$

Пусть $\vec{a}$ и $\vec{x}$ исходное по стилю и сгенерированное изображение соответственно, а их G^l и $\hat{G}^l$ признаковые описания их стиля на слое l, т.е. соответствующие матрицы Грамма (см. предыдущий раздел). Тогда функция потерь по стилю вычисляется аналогично функции потерь для генерации текстур в работе [Gatys et al., Nov 2015] и имеет вид:

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G^l_{ij} - \hat{G}^l_{ij})^2$$

$$\mathcal{L}_{style}(\vec{a}, \vec{x}) = \sum_{l=0}^{L} w_l E_l$$

Производная E_l по активациям слоя l может быть вычислена аналитически и имеет вид:

$$\frac{\partial \mathcal{L}_{style}}{\partial F^l_{i,j}} = \begin{cases} \frac{1}{N_l^2 M_l^2} ((F^l)^T (G^l - \hat{G}^l)_{j,i}, & F^l_{i,j} > 0 \\ 0, & F^l_{i,j} < 0 \end{cases}$$

Для создания картины смешивающей содержание одного изображения и стиль другого будем пропускать через архитектуру, представленную на Рис. 36 изображение белого шума и совместно минимизировать loss по контенту от изображения структуры и loss по стилю от изображения стиля. Формула всего loss будет выглядеть следующим образом:

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x}),$$

где α, β - веса для содержания и стиля соответственно. Несколько примеров стилизации изображения показана на Рис. 37. На Рис. 38 показано влияние отношения коэффицентов α/β и увеличения числа слоев для представления стиля.

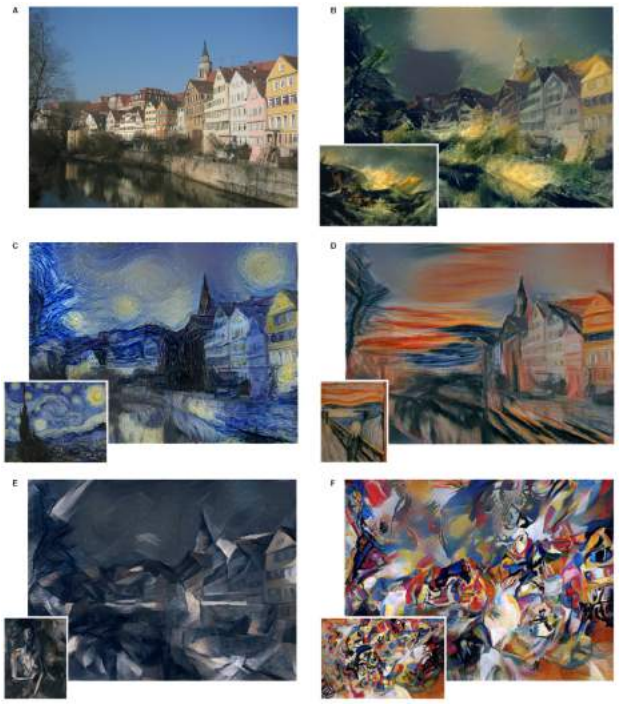

Рис. 37: Изображения,которые сочетают содержание фотографии со стилем нескольких известных произведений искусства. Изображения были созданы путем поиска изображения, которое одновременно соответствует представлению содержания фотографии и представлению стиля художественного произведения. **A** Оригинальная фотография, изображающая Neckarfront в Tubingen, Германия, показана на (Фото: Andreas Praefcke). Рисунок, задавший стиль для соответствующего сгенерированного изображения, отображается в левом нижнем углу каждой панели. **B** "The Shipwreck of the Minotaur"by J.M.W. Turner, 1805. **C** "The Starry Night"by Vincent van Gogh, 1889. **D** "Der Schrei"by Edvard Munch, 1893. **E** "Femme nue assise"by Pablo Picasso, 1910. **F** "Композиция VII"Василия Кандинского, 1913. [Gatys et al., Sep 2015]

III. Заключение

Представленные в данной работе методы помогают понимать и визуализировать, этапы работы глубоких нейронных сетей при решении задачи классификации, улучшать архитектуру сети и проверять, чему сеть научилась во время обучения. В данной работе рассмотрено несколько подходов для борьбы с основной проблемой при визуализации нейронной сети – высокочастоным шумом. Эти подходы позволяют генерировать качественные интерпретируемые изображения.

Генерация изображений используется не только для для визуализации нейронных сетей, но и в

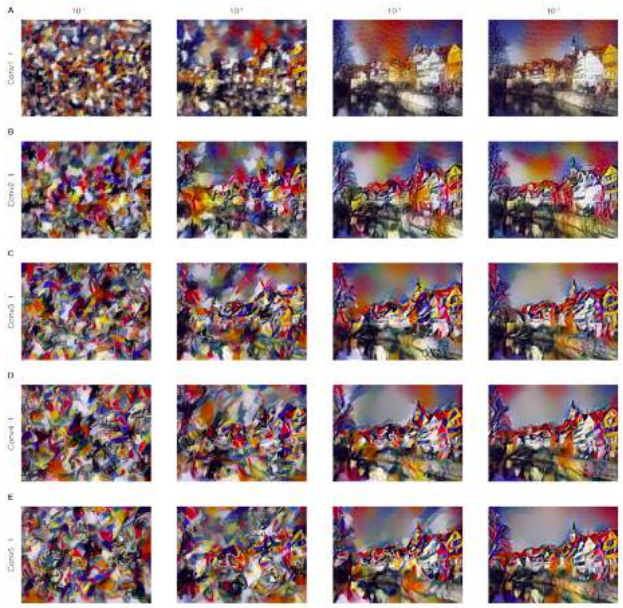

Рис. 38: Подробные результаты по стилю картины «Композиции VII» Василия Кандинского. Строки показывают результат сопоставления представления стиля при увеличении подмножеств слоев CNN. Локальные структуры изображений, захваченные представлением стиля, увеличиваются в размере и сложности при включении объектов стиля из более высоких слоев сети. Это можно объяснить увеличением размеров восприимчивых полей нейронов (receptive fields) и усложнением функций в иерархии обработки сети. Столбцы показывают различные отношения весов между реконструкцией содержания и стиля. Над каждым столбцом указано соотношение α/β между акцентом на соответствие содержанию фотографии и стилем художественного произведения. [Gatys et al., Sep 2015]

ряде таких задач, как создание изображений с заданным признаковым описанием, генерация текстур и пейзажей, а также в задаче переноса стиля.

Список литературы

[Simonyan et al., 2014] K. Simonyan, A. Vedaldi, A. Zisserman, “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps” arXiv preprint arXiv:1312.6034, 2014

[Nguyen et al., 2015] A. Nguyen, J. Yosinski, J. Clune, T. Fuchs, H. Lipson, “Understanding Neural Networks Through Deep Visualization” arXiv preprint arXiv:1506.06579, 2015

[Mordvintsev et al., 2015] A. Mordvintsev, C. Olah, M. Tyka., “Inceptionism: Going deeper into neural networks” Google Research Blog, 2015

[Øygard, 2015] Audun M. Øygard, “Visualizing GoogLeNet Classes” Google Research Blog, 2015

[Olah et al., 2017] C. Olah, A. Mordvintsev, L. Schubert, “Feature Visualization” https://distill.pub/2017/feature-visualization/, 2017

[Olah, et al., 2018] C. Olah, A. Satyanarayan, I. Johnson, S. Carter, L. Schubert, K. Ye, A. Mordvintsev, “The Building Blocks of Interpretability” https://distill.pub/2018/building-blocks/, 2018

[Tyka, 2016] M. Tyka, "Class visualization with bilateral filters". https://mtyka.github.io/deepdream/2016/02/05/bilateral-class-vis.html, 2016

[Nguyen et al., 2016] A.Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, J. Clune, “Synthesizing the preferred inputs for neurons in neural networks via deep generator networks” arXiv preprint arXiv:1605.09304, 2016

[Dosovitskiy et al., 2017] A. Dosovitskiy, J. Springenberg, M. Tatarchenko, T. Brox, “Learning to Generate Chairs, Tables and Cars with Convolutional Networks” arXiv preprint arXiv:1411.5928, 2017

[Gatys et al., Nov 2015] L. Gatys, A. Ecker, M. Bethge, “Texture Synthesis Using Convolutional Neural Networks” arXiv preprint arXiv:1505.07376, 2015

[Gatys et al., Sep 2015] L. Gatys, A. Ecker, M. Bethge, “A Neural Algorithm of Artistic Style” arXiv preprint arXiv:1508.06576, 2015

[Ulyanov, 2017] D. Ulyanov, "Fast Neuro Doodle". https://github.com/DmitryUlyanov/fast-neural-doodle, 2017

[Carter, et al., 2019] S. Carter, Z. Armstrong, L. Schubert, I. Johnson, C. Olah, “Activation Atlas” https://distill.pub/2019/activation-atlas/, 2019

[Goh, et al., 2021] G. Goh, N. Cammarata, C. Voss, S.Carter, M. Petrov, L. Schubert, A. Radford, C. Olah, "Multimodal Neurons in Artificial Neural Networks” https://distill.pub/2021/multimodal-neurons/, 2021