**К ПРЕЗЕНТЕ ПО FACE ID**

За последнее десятилетие технология распознавания лиц сделала большой шаг вперёд — и в то же время стала предметом очень многих споров и обсуждений. В интернете можно найти огромное количество заметок и статей о том, как устроено распознавание лиц, зачем его внедряют и насколько хорошо или плохо оно работает. И, как это всегда и бывает, во всём этом объёме информации весьма непросто разобраться и отделить правду от досужих домыслов, особенно, если не обладать соответствующим бэкграундом. Скажем, один автор может утверждать, что сегодняшние нейросети умеют совершенно безошибочно определять нужного человека в большой толпе, другой — приводить примеры курьёзных ляпов искусственного интеллекта, а третий — раскрывать секретные способы гарантированно обмануть алгоритм распознавания.

**Что на входе, что на выходе**

Погружаться в устройство системы распознавания лиц предлагаем постепенно. Для начала нам будет проще всего представлять её чёрным ящиком, который на вход принимает изображение (это может быть кадр из видео или фотография), а на выходе возвращает некоторый набор вещественных чисел, который «кодирует» лицо. Этот набор часто ещё называют «вектором признаков» (и он фактически является вектором) или «биометрическим шаблоном».

Размерность этого вектора у каждой системы может быть своя, обычно это некоторая степень двойки: 128, 256 или 512. Какой бы ни была размерность, норма вектора равна единице: для вектора a = (a\_1, a\_2 \dots a\_N) будет справедливо \sqrt{a\_1^2+a\_2^2 + \dots + a\_N^2}=1/ . о означает, что все возвращаемые системой векторы лежат на N-мерной гиперсфере, где N — размерность вектора. Представить себе такую гиперсферу довольно сложно, поэтому — опять-таки, для простоты — прибегнем к известному совету Джефри Хинтона о визуализации многомерного пространства: To deal with hyper-planes in a 14-dimensional space, visualize a 3-D space and say “fourteen” to yourself very loudly. Everyone does it. (Когда вам предстоит иметь дело с гиперплоскостями в 14-мерном пространстве, представьте его себе трёхмерным и громко скажите: «Четырнадцать!» Все так поступают.) Итак, пусть у нас есть привычная нам трёхмерная сфера и трёхмерные векторы на ней, которые, как мы помним, и есть закодированные лица:

Эти векторы обладают следующим свойством: если мы попробуем дважды закодировать одно и то же изображение лица, мы получим два одинаковых вектора — угол между ними будет равен нулю, а чем сильнее будут различаться лица, тем дальше друг от друга они будут лежать на сфере и тем больше будет угол между ними. Это означает, что для определения «похожести» двух лиц нам достаточно измерить угол между их векторами; удобнее всего в качестве меры схожести использовать косинус угла, а не сам угол (не забываем, что все вектора имеют норму, равную 1): similarity = \cos{\Theta} = a\_1b\_1+a\_2b\_2+ \dots + a\_Nb\_N

А для ещё большего удобства перепишем меру схожести таким образом, чтобы она изменялась на интервале [0; 1]: similarity = \frac{1 + \cos{\Theta}}{2}

Система распознавания лиц не может нам сказать, что на некоторой фотографии изображён условный Иванов И. И. (или наоборот, что на фото совсем не Иванов) — она работает по-другому. Мы можем взять реальное фото Иванова и при помощи системы построить для него вектор признаков. В дальнейшем этот вектор можно будет сопоставить с вектором исследуемого изображения и узнать меру их схожести.

**Детектор**

Теперь будем постепенно открывать чёрный ящик. Первым делом, получив на вход картинку, алгоритму нужно отыскать на ней лица людей. За это отвечает компонент, называемый детектором, и его задача — выделить области, в которых содержится нечто, напоминающее лицо. До некоторых пор эту задачу решали методом Виолы — Джонса или при помощи HOG-детекторов, однако сегодня нейронные сети вытеснили их практически повсеместно — они точнее, менее чувствительны к ракурсу съёмки (наклонам, поворотам и тому подобному) и куда более стабильны в своих предсказаниях, чем классические методы. Даже скорость работы, которая традиционно считалась ключевым преимуществом «классики», сегодня перестала быть проблемой для нейросетей: с тем огромным объёмом доступных данных для обучения и развитием вычислительных ресурсов можно без проблем подобрать такой размер нейросети, который удовлетворит вашим запросам.

**Нормализатор**

Лица, которые возвращает детектор, всё ещё пребывают в своём естественном положении: как-то повёрнуты, как-то наклонены — а ещё они все, конечно, разного размера. Чтобы на следующих этапах нам было проще их обрабатывать и сравнивать, стоило бы привести их к некоторому универсальному виду. Эту задачу решает компонент системы, который называется нормализатором.

В идеале, мы хотели бы работать только с фронтальными изображениями лица, для чего, в свою очередь, необходимо уметь преобразовывать любое полученное системой, изображение к фронтальному типу, причём преобразовывать просто и быстро, не прибегая к 3D-реконструкции и прочему «ракетостроению». Разумеется, магии не существует, и легко привести любое лицо к фронтальному невозможно, однако мы всё ещё можем попытаться получить изображение, максимально приближённое к фронтальному, — насколько это возможно для имеющейся картинки. В нашем распоряжении имеются три инструмента:

* scale: мы можем «приблизить» или «отдалить» лицо;
* rotation: мы можем повернуть лицо на любой угол в плоскости изображения;
* shift: мы можем сместить лицо на несколько пикселей влево или вправо, вверх или вниз.

Каждое из этих преобразований описывается матрицей 3×3. Перемножив все три матрицы, мы также получим матрицу 3×3 для суммарного преобразования — его необходимо применить к лицу для приведения к нужному нам виду:

Одним из способов определения преобразования может быть следующий: найти ключевые точки лица (центры глаз, кончик носа) и вычислить такую матрицу, в результате применения которой кончик носа окажется по центру изображения, а глаза выровняются на одном горизонтальном уровне. Способ довольно простой, однако, во-первых, он сильно зависит от качества детектирования ключевых точек, а во-вторых, нет гарантии, что описанные выше эвристики являются оптимальными для распознавания.

Альтернатива, как вы уже, вероятно, догадались — снова нейросеть. С её помощью мы можем предсказать итоговую матрицу преобразования напрямую, не отыскивая ключевые точки и не делая каких-либо предположений о расположении носа и глаз. Именно пример работы нейросетевого нормализатора показан на рисунке выше (точки мы нанесли просто для иллюстрации).

**Экстрактор**

Теперь, когда у нас есть нормализованное лицо, настало время строить вектор — этим занимается компонент, называемый экстрактором, основной элемент всей системы. Он принимает на вход картинки фиксированного разрешения — обычно 90–130 пикселей, такой размер позволяет соблюсти баланс между точностью работы алгоритма и его скоростью (картинка большего разрешения могла бы содержать больше полезной для распознавания информации, но и обработка её выполнялась бы дольше).

Экстракция вектора — завершающий этап пайплайна обработки лица, который можно схематически изобразить следующим образом:

**Обучение экстрактора**

Главное, чего мы ждём от хорошего экстрактора — чтобы он строил как можно более «близкие» векторы для схожих лиц и как можно более «далёкие» — для непохожих. Для этого экстрактор нужно обучить, а для обучения первым делом нам понадобится датасет — набор размеченных данных. Выглядеть он может примерно так:

То есть, у нас есть некоторое множество уникальных людей — «персон» (Персона k и Персона m — это разные люди, если m ≠ k ), и для каждой из них есть некоторое множество картинок. При этом мы точно знаем, на какой картинке какая персона.

Сколько персон нам нужно для обучения? И сколько фотографий для каждой персоны? Напрашивается очевидный ответ: чем больше, тем лучше. Самые крутые системы обучаются на датасетах в миллионы, а то и в десятки миллионов персон, а вот фотографий на каждую из них нам будет достаточно пяти–десяти (да и персона с единственной фотографией может оказаться полезной для обучения), но опять же: больше — лучше. В наши дни в интернете можно найти большое количество публичных датасетов, а иные исследователи собирают для обучения фотографии знаменитостей.

При формировании обучающей выборки следует принимать во внимание тот факт, что экстрактор (на самом деле, это справедливо для любой нейросети) всегда будет лучше работать на данных, похожих на те, на которых он учился. Если в нашем датасете будут только представители европеоидной расы, есть большой риск, что результаты для лиц азиатского типа или темнокожих рас нас разочаруют — а значит, нужно постараться сделать обучающую выборку как можно более разнообразной.

**Часто задаваемые вопросы**

На что похожи признаки, которые определяет нейросеть? Чем руководствуется алгоритм при построении вектора? Может ли вектор сам по себе что-то сказать нам о внешности человека? И — вишенкой на торте: на какие именно части лица обращает внимание алгоритм? Пожалуй, ни о чём другом нас не спрашивают так часто, как об этом. На самом деле, это всё очень хорошие вопросы, и они сильно интересуют не только обывателей, но и самих исследователей — разработчиков нейросетей.

Если попросить простого человека описать приметы некоторого лица, он наверняка назовёт разрез и цвет глаз, особенности причёски и растительности на лице, длину носа, изгиб бровей… Тренированный физиогномист (например, пограничник, который проверяет ваш паспорт в аэропорту, или криминалист, специализирующийся на портретной экспертизе) оценит расположение антропометрических точек и ключевые расстояния между ними. Так же и нейросеть: она, безусловно, «обращает внимание» на характерные особенности лица, однако нужно понимать, что каждое из чисел, составляющих вектор признаков, не отвечает за какую-то конкретную точку или черту лица. Мы не можем, взглянув на числовое представление вектора, указать, что вот этот его участок описывает глаза, а вот тот — форму носа. (Между тем, дальше в этой статье мы проиллюстрируем ряд экспериментов, которые раскроют, насколько сложно нейросети узнать человека, не видя, например, его глаз или рта).

Итак, для ручного анализа вектор непригоден. В то же время, существует немалое количество научных исследований, авторы которых пытаются восстановить внешность человека, располагая вектором признаков. Среди них упомянем работу учёных из Канады и США Vec2Face. Суть предлагаемого ими решения, если излагать её максимально упрощённо, такая: для генерации изображения из вектора признаков будем использовать специальным образом устроенную нейросеть, а датасет для её обучения получим, прогнав большое количество фотографий через экстрактор и сохранив полученные векторы.

Авторам удалось получить весьма приемлемый результат:

В самом нижнем ряду на иллюстрации показаны реальные фото, а над ними — картинки, которые удалось синтезировать из их векторов, используя разные методы восстановления. Посмотрите сами: изображения в третьем и четвёртом рядах довольно близки к реальности!

Однако нужно понимать, что применение описанного подхода на практике будет очень сильно затруднено. Во-первых, процесс разработки алгоритма восстановления достаточно сложен, требует серьёзных вычислительных ресурсов и значительных затрат времени. Во-вторых, для создания алгоритма исследователям потребуется неограниченный доступ к экстрактору — а от этого не так уж и сложно защититься. В-третьих, для каждого нового экстрактора нужно обучать модель восстановления заново, невозможно создать универсальный алгоритм, который будет способен «инвертировать» любой экстрактор. В-четвёртых (и, наверное, в самых главных), далеко не факт, что для какого-то конкретного экстрактора в принципе получится создать алгоритм восстановления хорошего качества.

**Оцениваем качество изображения**

Начнём с того, что наш детектор может ошибочно распознать лицо там, где его нет. Или распознать правильно, но изображение окажется плохого качества: очень маленького размера или повёрнутое под очень большим углом, или сильно смазанное, или очень тёмное, или всё это сразу. Тем не менее, раз лицо найдено, оно пройдёт весь пайплайн и получит вектор — вот только применить его будет нельзя. В случае с некачественным лицом вектор получается очень шумным и неточным, а при ложном срабатывании детектора он и вовсе не несёт какой-либо полезной информации, а значит, использовать такие векторы для сравнения нет никакого смысла.

Нам нужен какой-то механизм, который позволит передавать экстрактору только качественные картинки. В NtechLab мы используем дополнительную легковесную нейросеть — детектор качества, которая возвращает интегральный показатель в диапазоне [0; 1], позволяющий сделать вывод, насколько «хорошее» изображение выдал детектор. Если измеренный показатель качества будет ниже некоторого установленного порога, такое лицо будет отбраковано, а двигая этот порог, мы получаем возможность тонкой подстройки системы к фактическим условиям съёмки.

Оценка качества

Что ж, вот мы и победили современный искусственный интеллект в рамках нашего датасета! Осталось выяснить только одну деталь: насколько велика разница между проприетарным и публичным алгоритмом? Чтобы получить объективный ответ, первым делом нужно понять, как померить «качество» работы алгоритма.

В принципе, все возможные сценарии биометрического сопоставления можно свести к двум — верификации и идентификации:

* верификация (она же сопоставление 1:1) представляет собой сравнение двух образцов для исследования их принадлежности одному и тому же человеку. Верификация, в частности, выполняется, когда вы пытаетесь разблокировать смартфон при помощи изображения лица — здесь биометрическая система отвечает на вопрос, достаточна ли высока её уверенность в том, что предъявленное изображение принадлежит владельцу устройства;
* идентификация (она же поиск, она же сопоставление 1:N) подразумевает отбор из некоторого множества образцов-кандидатов тех, что предположительно принадлежат тому же человеку, что и представленный системе искомый образец. В качестве примера можно предложить систему контроля доступа, которая отпирает магнитный замок, когда «видит» на камере знакомое лицо.

В самом начале мы уже говорили, что биометрическая система не возвращает ответы вроде «да, это точно он» или «нет, это точно не он». Результатом сопоставления векторов будет показатель схожести, измеряемый на интервале [0; 1], и для его приведения к бинарному ответу «да/нет» нам нужно ввести пороговое значение. Если показатель схожести в результате некоторого сравнения окажется выше порога или равным ему, будем расценивать ответ системы как «да», а если ниже — как «нет».

Про показатель схожести нужно сделать несколько важных оговорок. Во-первых, нельзя принимать показатель схожести, который вернула система, за «качество» сопоставления. Нам приходилось сталкиваться с ситуациями, когда при сравнении нескольких систем заказчики говорили: «Система А оценила совпадение двух лиц на 0,78, а система Б — на 0,91, а раз это один и тот же человек, то система Б отработала лучше». Во-вторых, сам по себе показатель схожести не говорит вообще ни о чём. Например, у какой-то системы схожесть 0,78 может свидетельствовать об очень высокой уверенности в совпадении двух кандидатов, а другая может возвращать 0,85 при сравнении двух разных людей. В-третьих, нельзя трактовать схожесть как «совпадение» лиц — результат в 0,78 не означает, что «это один и тот же человек на 78%». Это просто число, которое должно интерпретироваться так или иначе в зависимости от выбранного порога.

Самым очевидным и самым наивным подходом к оценке качества алгоритма было бы измерение точности сопоставления (accuracy) — отношения случаев, когда система сработала верно, к общему количеству попыток сопоставления.

Accuracy = (Случаи, когда система сработала верно) / (Всего случаев)

В чём именно состоит его наивность и почему он недостаточно хорош? Давайте порассуждаем. Первое: даже умозрительно очень легко заключить, что, каким бы ни был установленный нами порог, всегда возможны только четыре исхода сравнения:

* истинно-положительный (true match): вычисленный показатель схожести выше установленного порога, и оба сравниваемых образца фактически принадлежат одному и тому же человеку;
* истинно-отрицательный (true non-match): вычисленный показатель схожести ниже установленного порога, и оба сравниваемых образца фактически принадлежат разным людям;
* ложноотрицательный (ошибка первого рода, false non-match): вычисленный показатель схожести ниже установленного порога, в то время как оба сравниваемых образца фактически принадлежат одному и тому же человеку;
* ложноположительный (ошибка второго рода, false match): вычисленный показатель схожести выше установленного порога, в то время как оба сравниваемых образца фактически принадлежат разным людям.

Второе: рассматривая accuracy в качестве единственной или главной метрики, затруднительно сделать вывод, какое количество ошибок допустила система. Допустим, мне всякий раз удавалось разблокировать мой собственный смартфон изображением лица (субъективная точность — 100%), но при этом скольким самозванцам и злоумышленникам удалось проделать с моим смартфоном то же самое?

Рассуждаем дальше: если мы опустим порог очень близко к нулю, то почти для любого кандидата мы всегда получим какое-то совпадение, но большинство из них будут ложными (будет много ошибок второго рода), а если поднимем его почти до единицы, получим только самые правильные совпадения, допустив при этом большое количество ошибок первого рода. Стало быть, увеличивая или уменьшая порог, мы можем получить более строгую или менее строгую систему. В случае со смартфоном большая строгость — это хорошо: пусть мне придётся сделать несколько попыток, но самозванцу разблокировать устройство будет очень сложно. И другое дело, когда мы пытаемся найти пропавшего без вести человека в огромном мегаполисе: пусть ложных совпадений будет и больше, зато мы точно не пропустим того, кого ищем.

Но как сравнить два алгоритма, если у одного порог настроен более строго, а у другого — менее? Прямое сопоставление метрик accuracy в этом случае оказывается совершенно бессмысленным! Выход прост: вместо того чтобы измерять частоту правильных сравнений, изучим, какова частота истинно-положительных результатов алгоритма при фиксированной частоте ошибок второго рода. Такая метрика обозначается как TMR@FMR=\alpha, где:

* TMR— true match rate, отношение числа истинно-положительных результатов к общему числу выполненных «позитивных» сравнений (когда оба сравниваемых образца фактически принадлежат одному и тому же человеку);
* FMR— false match rate, отношение числа ошибок второго рода к общему числу выполненных «негативных» сравнений (когда сравниваемые образцы фактически принадлежат разным людям);
* \alpha— фиксированное значение FMR, при котором замеряется TMR.

**IPHONE**

На iPhone и iPad хранятся многие составляющие нашей цифровой жизни, которые необходимо защищать. Технология Touch ID в свое время стала настоящим прорывом в области аутентификации по отпечатку пальца, а Face ID — это совершенно новый уровень распознавания пользователя по лицу. Face ID представляет собой интуитивный и надежный метод аутентификации, в основе которого лежит создание структурной карты лица с помощью инновационной системы камер TrueDepth и передовых технологий. Face ID — это результат объединения самых передовых аппаратных и программных компонентов Apple. Камера TrueDepth захватывает данные лица, проецируя на него и анализируя несколько тысяч невидимых точек. Таким образом устройство составляет подробную структурную карту лица, а также его изображение в инфракрасном спектре. Фрагмент системы Neural Engine микропроцессоров A11, A12 Bionic, A12X Bionic, A13 Bionic, A14 Bionic и A15 Bionic, защищенный модулем Secure Enclave, преобразует структурную карту и инфракрасное изображение в математическое представление, которое сравнивается с зарегистрированными данными лица. Технология Face ID автоматически адаптируется к таким изменениям внешности, как макияж или небритость. При более значительных изменениях (например, сбривание бороды) система предложит ввести пароль, чтобы подтвердить вашу личность, и только после этого обновит данные о лице. Технология Face ID распознает лица даже при наличии шляпы, шарфа, контактных линз, корректирующих и солнцезащитных очков. Кроме того, она работает в помещениях, на улице и даже в полной темноте. В iOS 15.4 и более поздних версиях и на iPhone 12 или более новых моделях функцию Face ID можно использовать даже не снимая маску. Вероятность того, что другой человек сможет разблокировать ваш iPhone или iPad Pro с помощью Face ID, составляет менее чем 1 из 1 000 000, если зарегистрирован только один внешний вид (в маске или без нее). В качестве дополнительной меры безопасности допускается только пять неудачных попыток разблокировки с помощью Face ID, после чего будет запрошен пароль. Для близнецов, похожих друг на друга братьев и сестер, а также для детей младше 13 лет, у которых могли не до конца сформироваться характерные черты лица, статистическая вероятность выше, а при использовании Face ID в маске она возрастает еще больше. Если вы опасаетесь случайной разблокировки устройства в каком-либо из этих случаев, рекомендуем использовать для аутентификации пароль. Использовать функцию Face ID можно и без включения режима ношения маски.

Технология Face ID проверяет соответствие с использованием данных о структуре лица, которые невозможно считать с напечатанной или цифровой двухмерной фотографии. А сложные нейронные сети защищают от мошенничества с использованием масок или других приемов. Технология Face ID даже определяет осознанность ваших действий, поэтому при наличии на лице маски всегда будет выполняться проверка на внимание. Face ID сработает, только если ваши глаза будут открыты и вы будете смотреть на устройство. Это усложняет задачу разблокировать устройство без вашего ведома (например, когда вы спите). Компания Apple заботится о конфиденциальности данных своих клиентов. Данные Face ID (в том числе математические представления лица) шифруются и защищаются с помощью модуля Secure Enclave. Эти данные будут уточняться и обновляться по мере использования Face ID, в том числе при успешной аутентификации. Кроме того, Face ID обновит данные, если обнаружит близкое совпадение, после которого устройство будет разблокировано с помощью пароля. Данные Face ID хранятся на устройстве и никогда не копируются в iCloud или куда-либо еще.