Машинное обучение (Machine Learning)

One-shot learning, Siamese networks и zero-shot learning

Уткин Л.В.

Санкт-Петербургский политехнический университет Петра Великого

One-Shot Learning

Пример

Формальная постановка задачи

Дано:

- малое “помеченное” обучающее множество S из N примеров одинковой размерности с метками y

$$S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_N, y_N)\}$$

- тестовый пример $\widehat{\mathbf{x}}$, который нужно классифицировать

Цель:

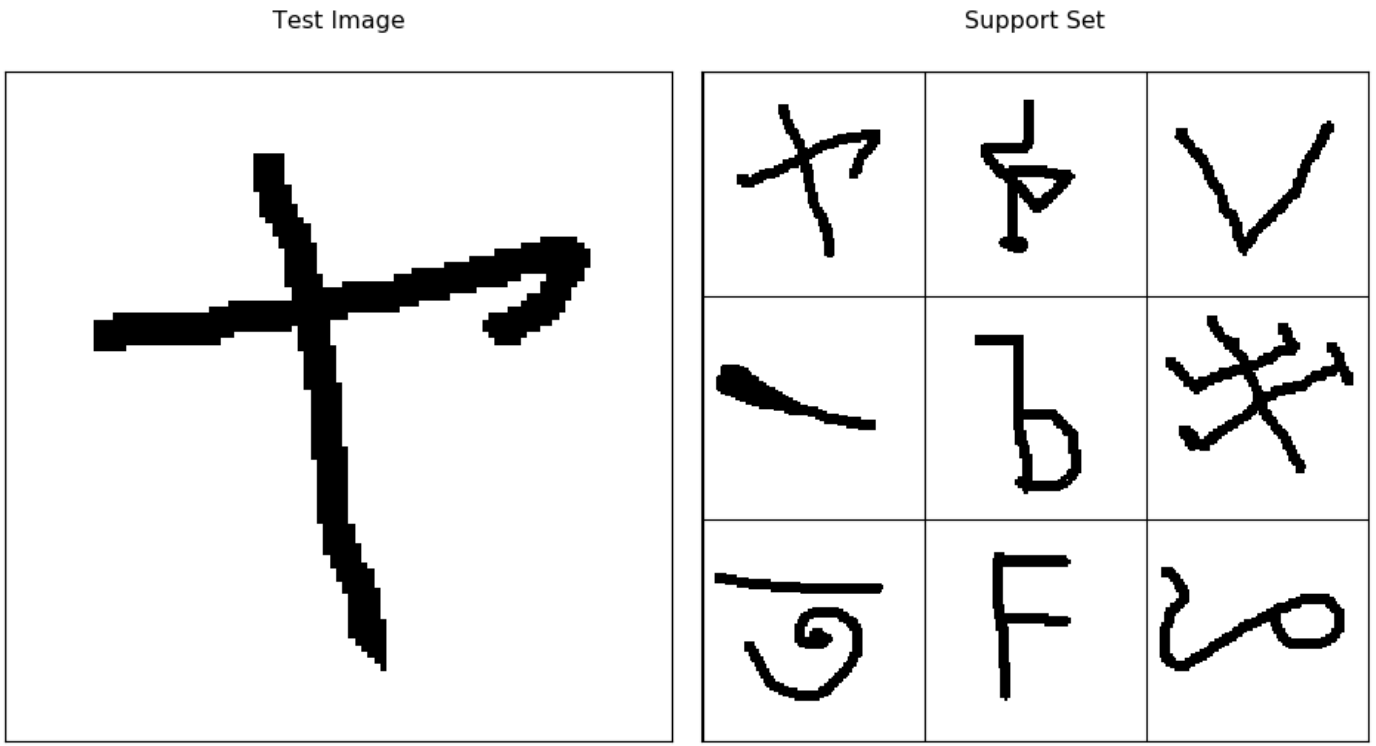
- так как ровно один пример имеет “правильный” класс, то необходимо определить $y \in S$ такое же как метка $\widehat{y}$ примера $\widehat{\mathbf{x}}$

Что нужно учесть при решении

- В реальности не всегда есть ограничение, что только одно изображение имеет правильный класс
- Просто обобщить эту ситуацию на случай k-shot, если есть не один, а k примеров для каждого y_i, а не один.
- Когда N большое, есть большее число возможных классов, к которым может принадлежать $\widehat{\mathbf{x}}$, поэтому сложнее предсказать правильный класс.
- Случайное угадывание будет иметь $\frac{100}{N}\%$ точность в среднем

Примеры

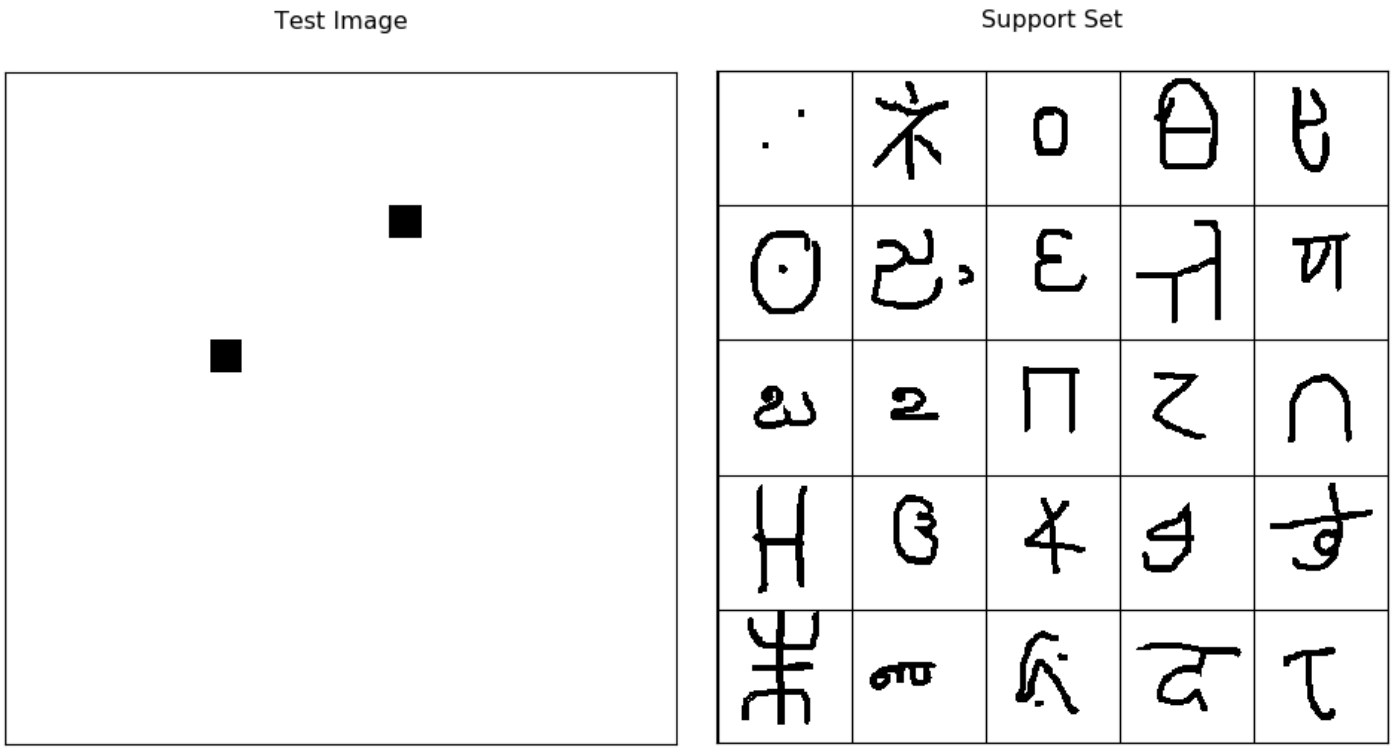
Датасет Omniglot $N = 9$



Датасет Omniglot представляет собой набор из 1623 рисованных символов в разрешении 105x105 из 50 алфавитов.

Примеры

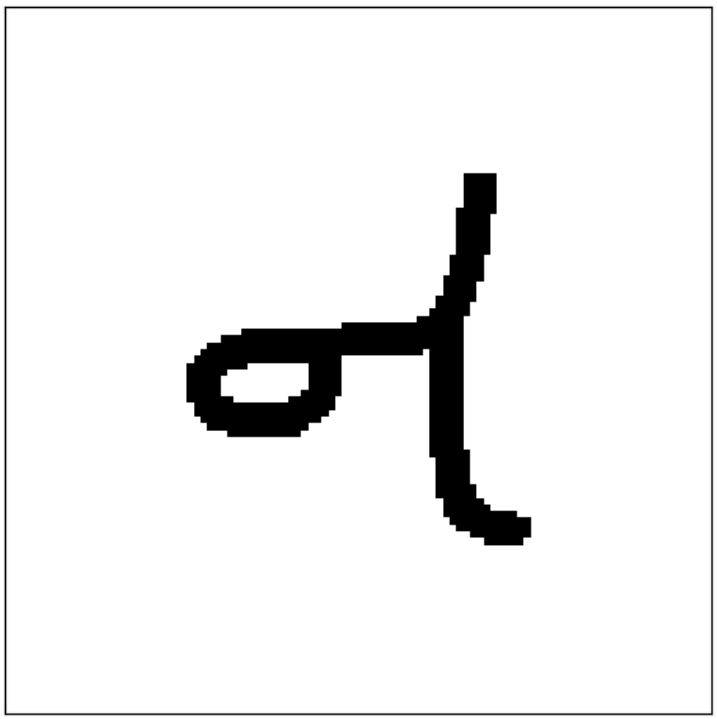
Датасет Omniglot $N = 25$



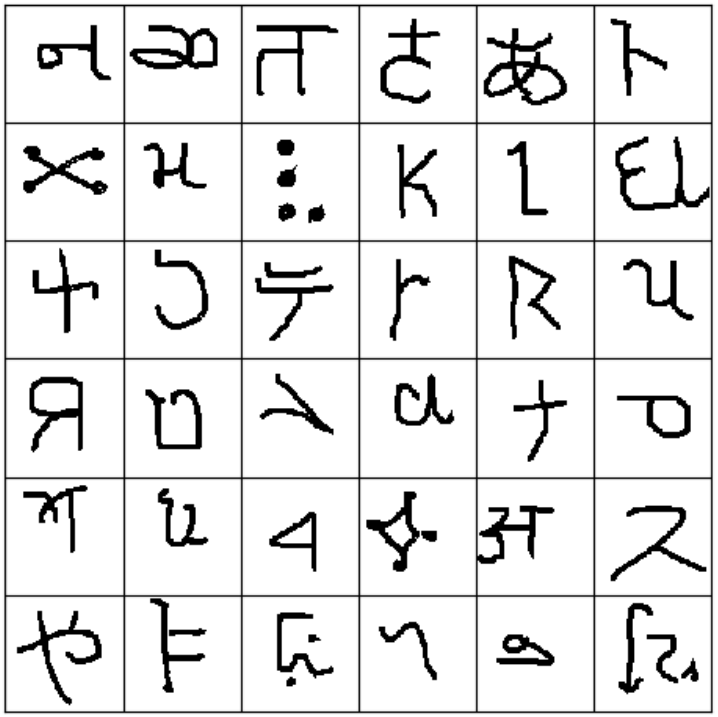
Примеры

Датасет Omniglot $N = 36$

Test Image



Support Set



Omniglot

Sanskrit

Greek

Bengali

Простейший метод классификации - 1 ближайший сосед

- Простейший способ классификации - это k ближайших соседей, но поскольку для каждого класса есть только один пример, используем 1 ближайшего соседа.
- Евклидово расстояние от тестового примера до обучающего:

$$C(\widehat{\mathbf{x}}) = \arg\min_{c \in S} \|\widehat{\mathbf{x}} - \mathbf{x}_c\|$$

- Точность (Koch и др.): $\sim 28\%$ при $N = 20$ omniglot
- Это примерно в 6 раз больше, чем просто случайное угадывание (5%)
- У людей точность 95.5% при $N = 20$ omniglot
- *Hierarchical Bayesian Program Learning* (Lake и др.) дает 95.2%

Неронные сети для обучения

- Как обучить нейронную сеть на единичных примерах? Переобучение!
- Многие подходы используют Transfer Learning
- Вспомним 1 ближайшего соседа - просто классифицирует путем поиска ближайшего примера на расстоянии L_2 (Евклидово расстояние)
- Но эта метрика плоха для большой размерности

Сиамские сети

Сиамские сети

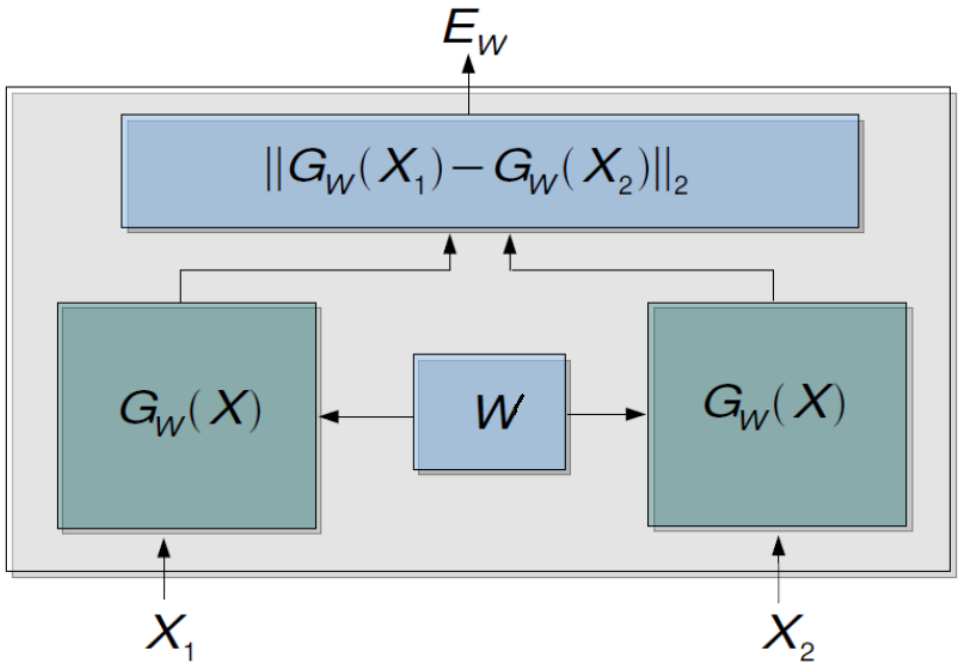
Идея: Сиамская сеть может сравнивать тестовое изображение с каждым изображением в наборе и выбирать, какое из них, имеет один и тот же класс - наиболее близко.

Элементы сиамских сетей

- X_1 и X_2 - пара изображений
- $Y = 0$, если X_1 и X_2 - один объект, $Y = 1$, если X_1 и X_2 - различны
- Построить нейронную сеть с минимальным числом параметров, определяющую для пар объектов, одинаковы ли она или нет

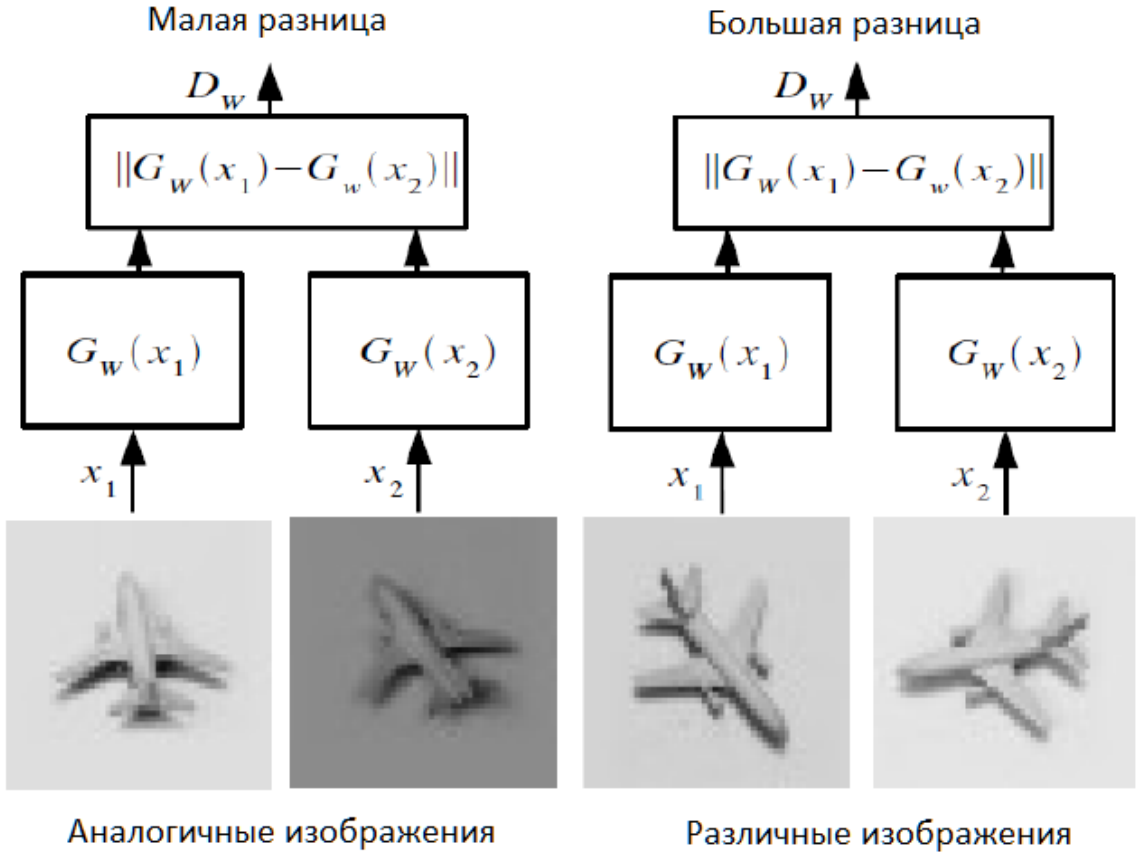
Архитектура сиамских сетей

Y. LeCun. Learning Hierarchies of Invariant Features



- W - общий вектор параметров,
- $G_W(X_1)$, $G_W(X_2)$ - точки в прост-ве меньшей размерности
- E_W - функция совместимости между X_1 и X_2 (“энергия”)

Еще пример сиамских сетей



Y. LeCun. Learning Hierarchies of Invariant Features

Функция потерь

- Функция потерь зависит от входных данных и параметров косвенно через энергию:

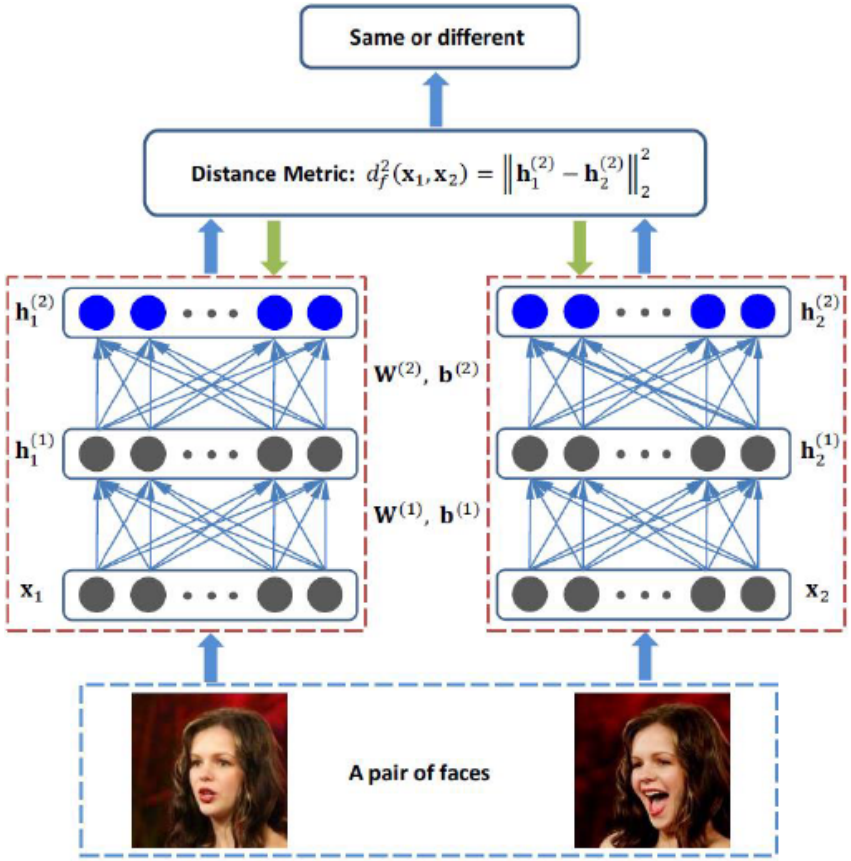
$$\mathcal{L}(W) = \sum_{i=1}^{N} L\left(W, (Y, X_1, X_2)_i\right)$$

$$L\left(W, Y, X_1, X_2\right) = (1-Y)L_G\left(E_W(X_1, X_2)\right) + YL_I\left(E_W(X_1, X_2)\right)$$

$$E_W = \|G_W(X_1) - G_W(X_2)\|$$

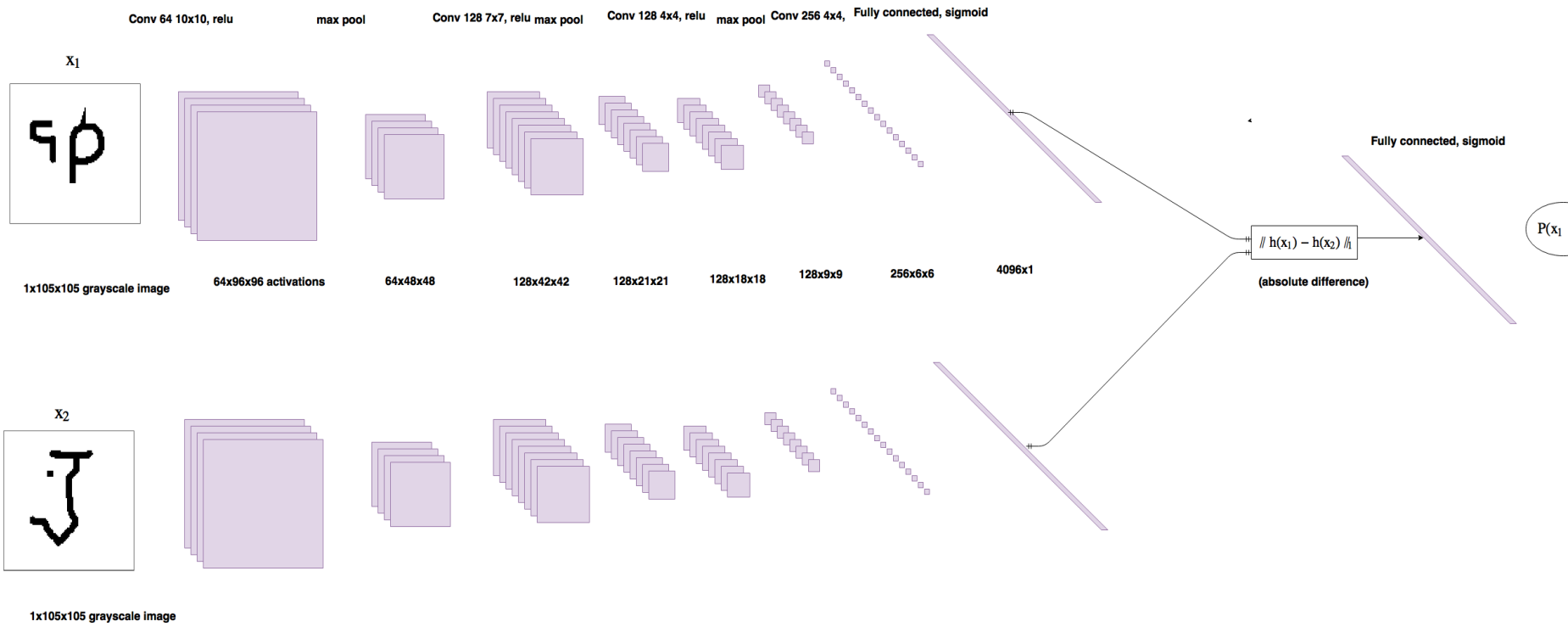
- L_G - функция потерь для совпадающих пар $Y = 0$
- L_I - функция потерь лоя несовпадающих пар $Y = 1$

Применение к распознаванию лиц



JunlinHu, etc. Discriminative Deep Metric Learning for Face Verification in the Wild, CVPR 2014

Глубокая сиамская сеть



Глубокая сиамская сеть

- Используем $t = 1$, если два изображения одного класса и $t = 0$ иначе
- Функция потерь

$$\begin{aligned} L(\mathbf{x}_1, \mathbf{x}_2, t) = & \; t \cdot \log(p(\mathbf{x}_1 \circ \mathbf{x}_2)) \\ & + (1 - t) \cdot \log(1 - p(\mathbf{x}_1 \circ \mathbf{x}_2)) \\ & + \lambda \cdot \|w\|_2 \end{aligned}$$

- Решение

$$C(\hat{\mathbf{x}}, S) = \arg\max_c P(\hat{\mathbf{x}} \circ x_c), \;\; x_c \in S$$

Глубокая сиамская сеть - обучение

- Почему нет переобучения
- Если есть C примеров в E классах, то число пар среди $C \cdot E$ примеров $N_{\text{пар}} = C \cdot E \cdot (1 - C \cdot E)/2$
- 20 примеров Omniglot из 964 классов - 185 849 560 пар!
- Но число примеров одного класса $N_{\text{одинак}} = \binom{E}{2} C$.

  Это 183 160 пар.
- Важно: для обучения сиамской сети необходимо соотношение 1 : 1 примеров одного и разных классов

Характеристики

https://sorenbouma.github.io/blog/oneshot/

Zero-shot learning

- Zero-shot learning позволяет модели распознавать то, что она раньше не видела.
- Представим, что нужно разработать модель машинного обучения, которая может классифицировать **всех** животных.
- Для этого нужен размеченный набор данных, по крайней мере, с одним примером для каждого отдельного животного.
- Это не помогает, когда существует огромное количество разных классов (например, видов животных), которые модель должна выучить.

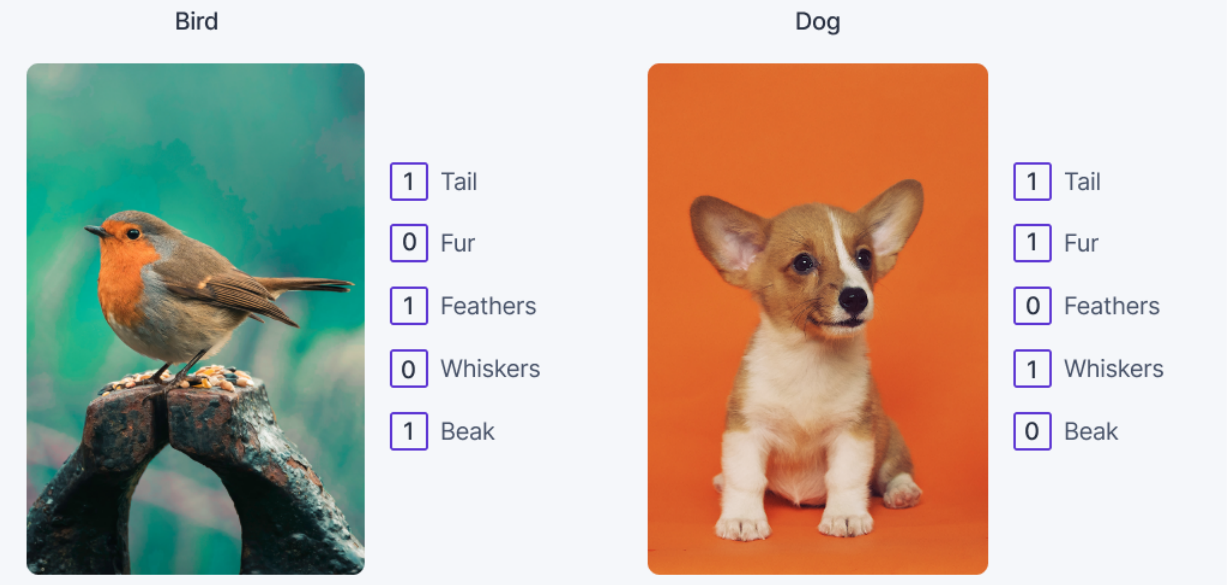
Zero-shot learning

- Один из способов - уменьшить зависимость моделей от размеченных данных. Это мотивация zero-shot learning, при котором модель учится классифицировать классы, которых она раньше не видела.
- Общая идея обучения zero-shot learning состоит в том, чтобы передавать знания, уже содержащиеся в учебных примерах в задачу классификации примеров. Таким образом, zero-shot learning является частью Transfer Learning.

Zero-shot learning

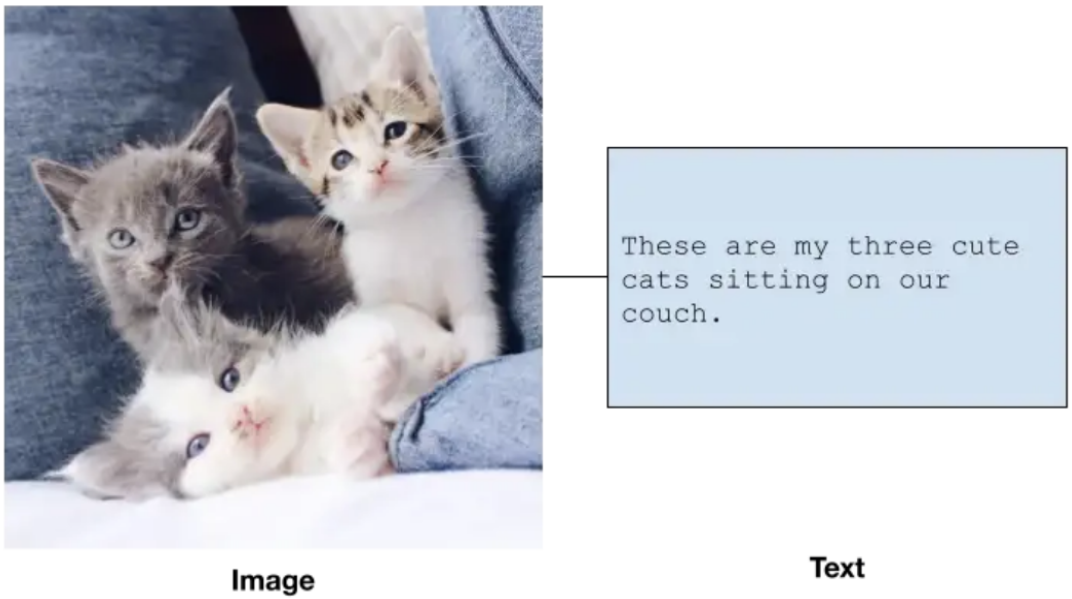
- В zero-shot learning данные состоят из следующих множеств:
- **Seen Classes**: это классы данных, которые использовались для обучения модели глубокого обучения.
- **Unseen Classes**: это классы данных, на которые необходимо обобщить существующую глубокую модель. Примеры этих классов не использовались при обучении.
- **Auxiliary Information**: поскольку размеченные примеры из Unseen Classes недоступны, для решения zero-shot learning необходима дополнительная информация. Она должна содержать информацию обо всех Unseen Classes, которая может быть описанием, семантической информацией или эмбедингами слов.

Auxiliary Information



Пример семантического эмбединга, использующего вектор атрибутов

Auxiliary Information еще



Auxiliary Information?

- Можно задаться вопросом, а не является ли этот “вспомогательный текст” типом метки?
- В то время как вспомогательная информация (т.е. подписи) является формой “учителя”, она не является меткой.
- Благодаря этой вспомогательной информации мы можем использовать богатые информацией неструктурированные данные вместо того, чтобы самостоятельно их анализировать, чтобы вручную создать одну метку (например, “это мои три милых кота...” Ž192 “кошки”).
- Разметка требует времени и сокращает потенциально полезную информацию.

Auxiliary Information?

- Люди могут естественным образом находить сходство между классами данных, например. заметив, что и у кошек, и у собак есть хвосты, и те, и другие ходят на четырех ногах и т.д.
- И Seen Classes, и Unseen Classes связаны в многомерном векторном пространстве, называемом *семантическим* пространством, где знания из Seen Classes могут быть перенесены в Unseen Classes.

Два подхода zero-shot learning

- Два наиболее распространенных подхода, используемых для решения проблем zero-shot learning:
  - методы на основе классификации
  - методы на основе примеров

Методы на основе классификации

- **Методы соответствия (Correspondence Methods)**: направлены на создание классификатора для unseen classes путем соответствия между бинарным классификатором “один против остальных” для каждого класса и его соответствующим прототипом класса.
- **Методы отношений (Relationship Methods)**: направлены на создание классификатора unseen classes на основе их меж- и внутриклассовых отношений для unseen classes. отношения между unseen classes и seen classes могут быть получены путем вычисления отношений между соответствующими прототипами.

Методы на основе классификации

- **Комбинированные методы (Combination Methods)**: описывают идею построения классификатора для unseen classes объединением классификаторов для основных элементов, используемых для создания классов. Считается, что есть список “основных элементов” для формирования классов. Каждая точка в seen и unseen classes есть комбинация этих основных элементов. Например, образ класса “собака” будет иметь хвост, мех и т. д. В семантическом пространстве, считается, что каждое измерение представляет собой базовый элемент, а каждый прототип класса обозначает комбинацию этих элементов. Каждое измерение прототипов классов принимает значение 1 или 0, обозначая, имеет ли класс соответствующий элемент.

Методы на основе примеров

- Методы на основе примеров нацелены сначала на получение размеченных примеров для unseen classes, а затем с помощью этих примеров на обучение zero-shot классификатора. В зависимости от источника этих примеров методы можно разделить на три подкатегории:
  - методы проекции (Projection methods)
  - методы заимствования примеров (Instance-borrowing methods)
  - методы синтеза (Synthesizing Methods)

Методы на основе примеров

- **Методы проекции (Projection methods)**: Идея - получить размеченные экземпляры для невидимых классов путем проецирования как примеров пространства признаков, так и прототипов семантического пространства в общее пространство. В пространстве признаков есть размеченные обучающие примеры, принадлежащие наблюдаемым классам. Прототипы также можно рассматривать как помеченные примеры. Таким образом, мы разметили примеры в двух пространствах (признаковом и семантическом пространствах).

Методы на основе примеров

- **Методы заимствования примеров (Instance-borrowing methods)**: получают размеченные примеров для невидимых классов путем заимствования из обучающих примеров. Основаны на сходстве классов. Пусть строим классификатор для класса “грузовик”, но у нас нет соот-щих размеченных примеров. Однако у нас есть размеченные примеры, принадлежащие классам “автомобиль” и “автобус”. Они аналогичны “грузовику”. При обучении класс-ров для класса “грузовик” используем примеры из этих двух классов. Это как люди распознают объекты. Мы никогда не видели примеры из некоторых классов, но видели примеры из похожих классов. Зная эти похожие классы, можем распознавать примеры, из невидимого класса.

Методы на основе примеров

- **Методы синтеза (Synthesizing Methods)**: Идея - получить размеченные примеры для невидимых классов путем синтеза псевдопримеров с использованием различных стратегий. В некоторых методах для синтеза псевдопримеров предполагается, что примеры каждого класса следуют некоторому распределению. Во-первых, необходимо оценить параметры распределения для невидимых классов. Затем синтезируются примеры невидимых классов.

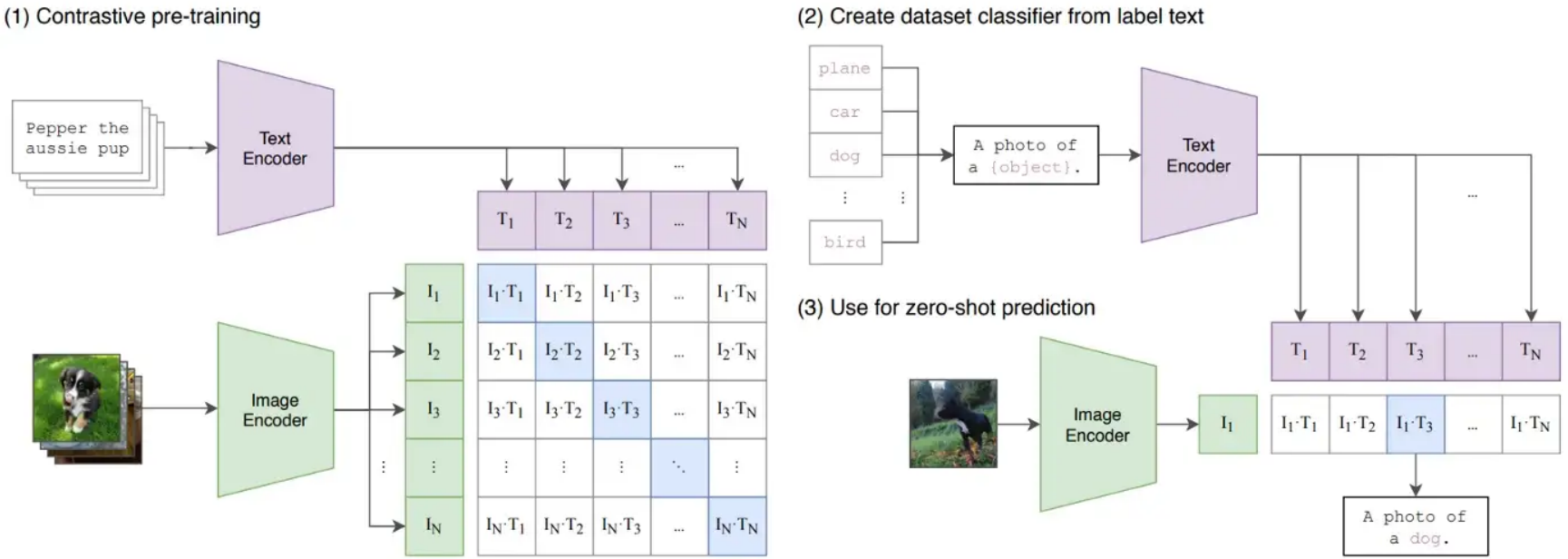
Как работает zero-shot learning

- Один из методов - Contrastive Language-Image Pretraining (CLIP), предложенный OpenAI
- Интуиция: CLIP состоит из двух этапов: этапа обучения (обучения) и этапа вывода (предсказания).
- На этапе обучения CLIP узнает об изображениях, “читая” вспомогательный текст (т.е. предложения), соответствующий каждому изображению.

Как работает zero-shot learning

- Как человек (при условии, что вы никогда раньше не видели кошку), вы можете прочитать этот текст и, возможно, расшифровать, что три вещи на изображении — это “кошки”. Если вы видели достаточно фотографий кошек с подписями со словом “кошка”, скорее всего, вы действительно хорошо определили, есть ли кошки на изображении.
- Точно так же, увидев 400 миллионов пар изображений и текста различных объектов, модель способна понять, как определенные фразы и слова соответствуют определенным шаблонам на изображениях. Получив это понимание, модель может использовать накопленные знания для экстраполяции на другие задачи классификации.

Как работает zero-shot learning



Zero-shot training (1)

- Как именно модель может учиться на этих вспомогательных текстах?
- Как следует из названия, CLIP использует метод, называемый контрастным обучением, чтобы понять взаимосвязь между парами изображения и текста.
- По сути, CLIP стремится свести к минимуму разницу между кодировкой изображения и соответствующего ему текста. Другими словами, модель должна научиться делать кодировки изображений и кодировки соответствующего им текста как можно более похожими.

Zero-shot training (2)

- Давайте разберем эту идею еще немного.
- Что такое кодировки? Кодирование — это просто низкоразмерное представление данных (зеленый и фиолетовый прямоугольники на рисунке выше). В идеале кодировка изображения или текста должны представлять наиболее важную и различимую информацию об этом изображении или тексте.
- Например, все изображения кошек должны иметь одинаковые кодировки, поскольку все они содержат кошек, но при этом они должны отличаться от кодировок собак.

Zero-shot training (2)

- В этом идеальном мире, где кодировки похожих объектов также схожи, а кодировки разных объектов также различны, становится очень легко классифицировать изображения. Если мы загрузим изображение в модель, и кодировка похожа на некоторые другие кодировки “кошки”, которые модель видела, она может сказать, что это “кошка”!
- Ключом к хорошей классификации изображений является изучение идеальных кодировок изображений. На самом деле это вся предпосылка CLIP. Мы начинаем с ужасных кодировок (т. е. случайных кодировок для каждого изображения) и хотим, чтобы модель выучила идеальные кодировки (т. е. изображения кошек имеют похожие кодировки).

Zero-shot training (3)

- Почему кодировка изображения должна быть максимально похожа на соответствующую кодировку текста? Теперь, когда мы знаем, что такое кодировка и почему важно выучить хорошие кодировки, мы можем понять, почему мы заставляем модель делать кодировку изображения и текста одинаковой.
- Напомним, что наша конечная цель — научиться классифицировать изображения, и поэтому нам необходимо научиться правильному представлению изображений (кодированию). Когда у нас были метки, мы могли научиться хорошим кодировкам, сводя к минимуму разницу между выходом модели и меткой.

Zero-shot training (4)

- Однако в случае CLIP у нас нет отдельных выходов модели. Вместо этого мы можем обрабатывать кодировки обучающих изображений как выходные данные модели, а текстовые кодировки соответствующих заголовков — как ожидаемые выходы.
- На самом деле мы можем представить, что **модель учится создавать хорошие метки**. Поскольку кодировщик текста также обновляется в процессе, со временем модель учится извлекать из текста более важную информацию, тем самым обеспечивая нам лучшее кодирование текста (ожидаемый результат).

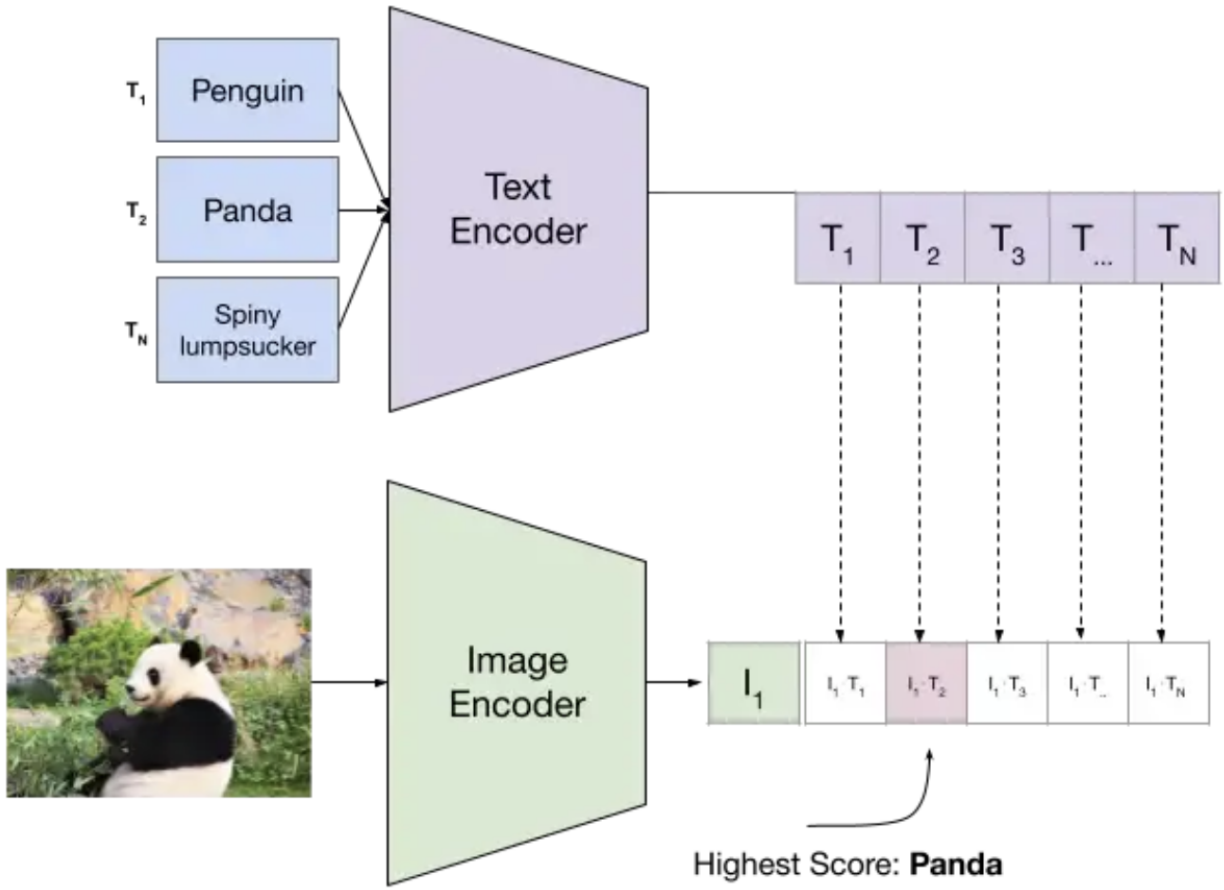
Zero-shot training (5)

- Имея это в виду, имеет смысл попытаться свести к минимуму разницу между кодировкой изображения и текста. *Это потому, что мы знаем, что похожие изображения, скорее всего, будут иметь одинаковую текстовую кодировку*, точно так же, как с метками, где похожие изображения будут иметь одинаковую метку. В результате модель научится генерировать одинаковые кодировки для похожих изображений.

Zero-Shot Inference (1)

- Как только модель обучена на достаточном количестве пар изображений и текста, ее можно использовать для вывода (для предсказания невидимых классов).
- На этапе вывода используем типовую задачу классификации, сначала получая список всех возможных меток. Поэтому, если бы мы предсказывали виды животных, нам понадобился бы список всех видов животных (например, пингвины, панды и т. д.)

Zero-Shot Inference (2)



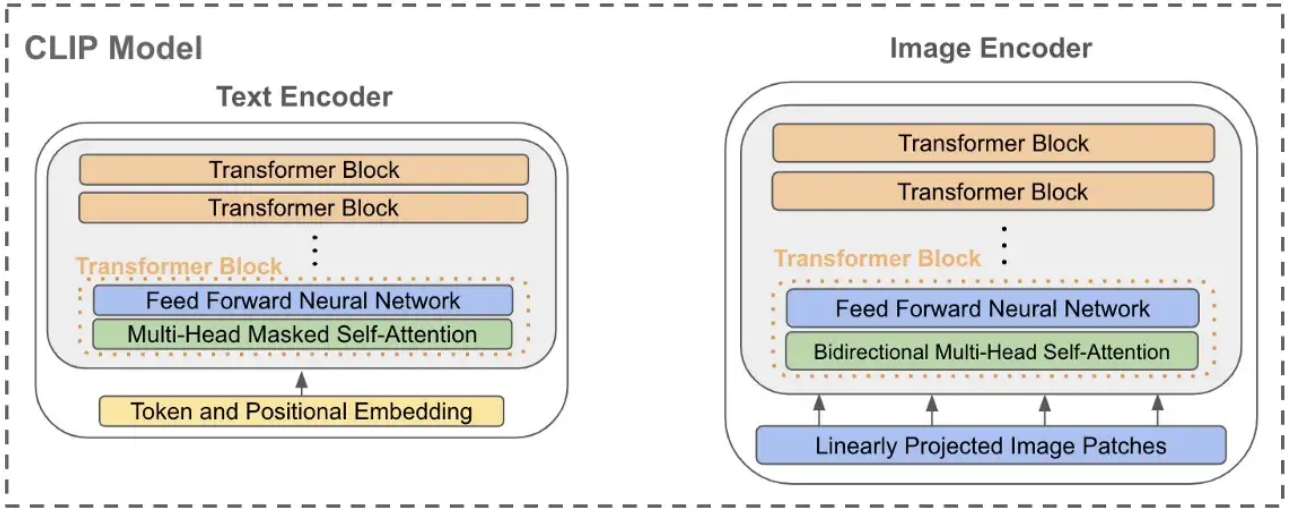
Zero-Shot Inference (3)

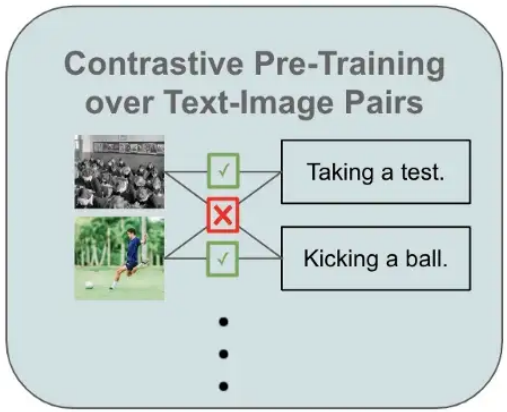
- Затем каждая метка будет закодирована предварительно обученным текстовым кодировщиком из шага 1.
- Теперь, когда у нас есть кодировки меток, от T_1 до T_n, можем взять изображение, которое хотим классифицировать, передать его через предварительно обученный кодировщик изображений и вычислить, насколько кодировка изображения похожа на кодировку каждой текстовой метки, используя метрику расстояния.

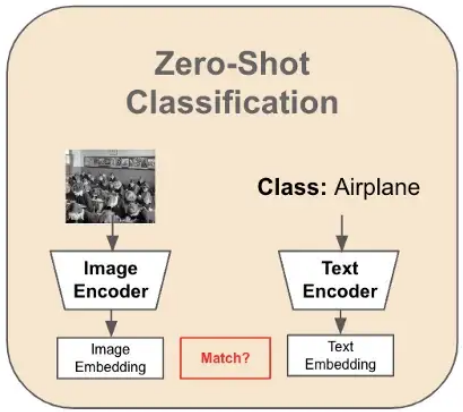
Zero-Shot Inference (4)

- Теперь мы классифицируем изображение как метку с наибольшим сходством с изображением. Мы можем сделать это, поскольку знаем, что модель научилась генерировать кодировки для изображений, максимально похожие на текстовый аналог, большинство из которых, вероятно, содержало метку, которую мы пытаемся классифицировать.
- Независимо от того, какой подход используется, общей темой zero-shot learning является то, что мы можем использовать некоторую вспомогательную информацию (например, текстовые описания), которая не является явными метками, в качестве слабой формы supervision.

Архитектура CLIP



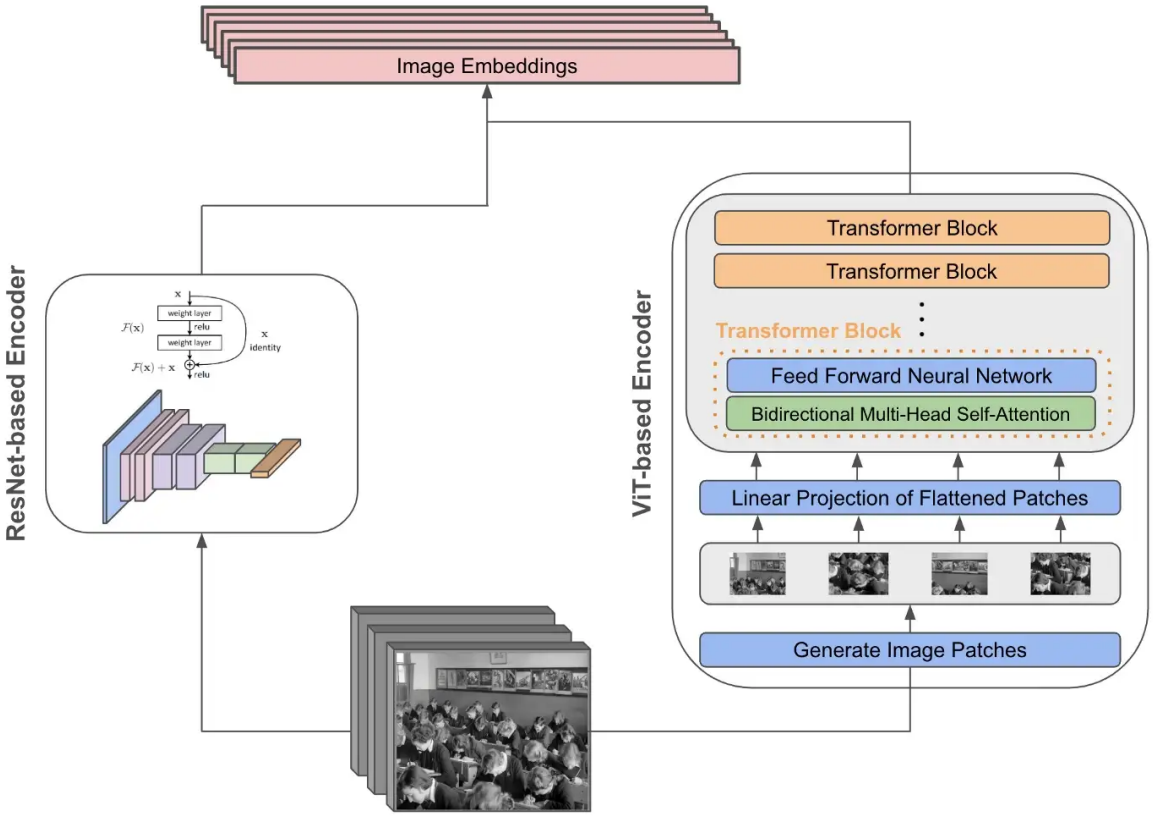




Архитектура CLIP

- CLIP состоит из двух модулей кодера, которые используются для кодирования текстовых и графических данных соответственно.
- Для кодировщика изображений исследуется множество различных архитектур моделей, в том числе пять ResNets разных размеров и три архитектуры vision transformer.
- Однако вариант CLIP с vision transformer в 3 раза более эффективен в вычислительном отношении для обучения, что делает его предпочтительной архитектурой кодировщика изображений.

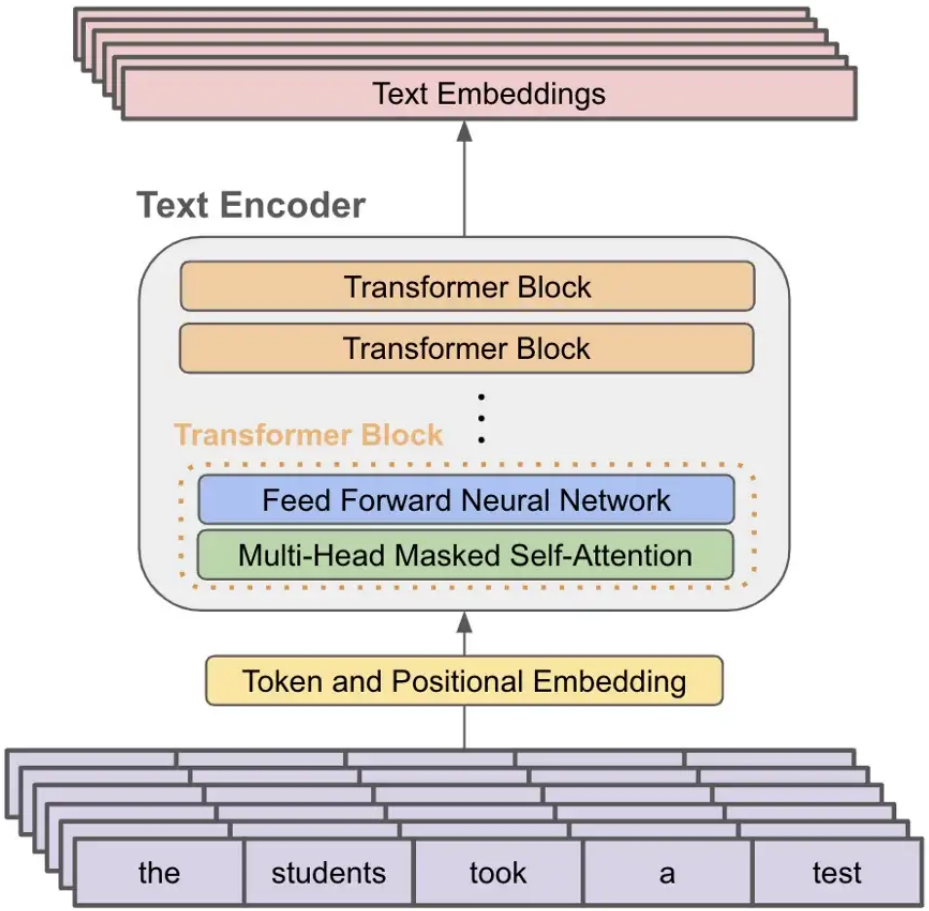
Архитектура CLIP



Архитектура CLIP

- Кодер текста в CLIP представляет собой просто decoder-only transformer, что означает, что маскированное самовнимание используется на каждом уровне.
- Маскированное самовнимание гарантирует, что представление преобразователя для каждого токена в последовательности зависит только от токенов, которые идут перед ней, тем самым не позволяя какому-либо токену смотреть “в будущее”, чтобы лучше информировать свое представление.
- Эта архитектура очень похожа на большинство ранее предложенных архитектур языкового моделирования (например, GPT-2 или OPT).

Кодер текста в CLIP



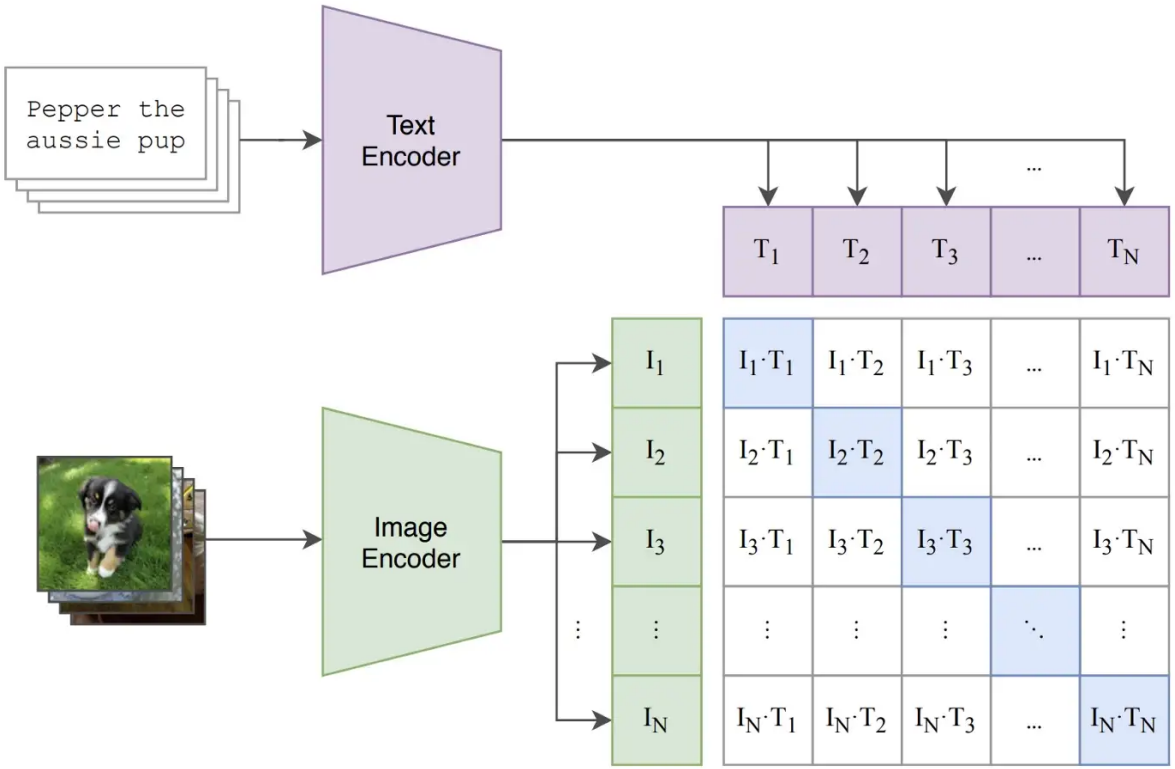
Обучение при помощи Natural Language Supervision

- Должны ли мы классифицировать изображения на основе слов в их подписи?
- Как насчет использования языкового моделирования для генерации подписи к каждому изображению?
- Интересно, что авторы CLIP считают, что предсказать точную подпись к изображению слишком сложно, что приводит к очень медленному обучению модели, из-за большого разнообразия способов описания любого изображения.

Обучение при помощи Natural Language Supervision

- Идеальная задача предобучения для CLIP должна быть масштабируемой, т.е. она позволяет модели эффективно обучать полезные представления из меток естественного языкы.
- CLIP можно эффективно обучать с помощью очень простой задачи - предсказания правильного связанного заголовка в группе подписей-кандидатов. Такая задача показана на рисунке далее.

Image-text contrastive pre-training



https://arxiv.org/pdf/2103.00020.pdf

Image-text contrastive pre-training

- На практике эта цель реализуется за счет:
  - прохождения группы изображений и текстовых подписей через соответствующие кодировщики
  - максимизации косинусного сходства между эмбедингами изображений и текстов истинных пар изображение-заголовок
  - минимизации косинусного сходства между всеми остальными парами изображений и подписей

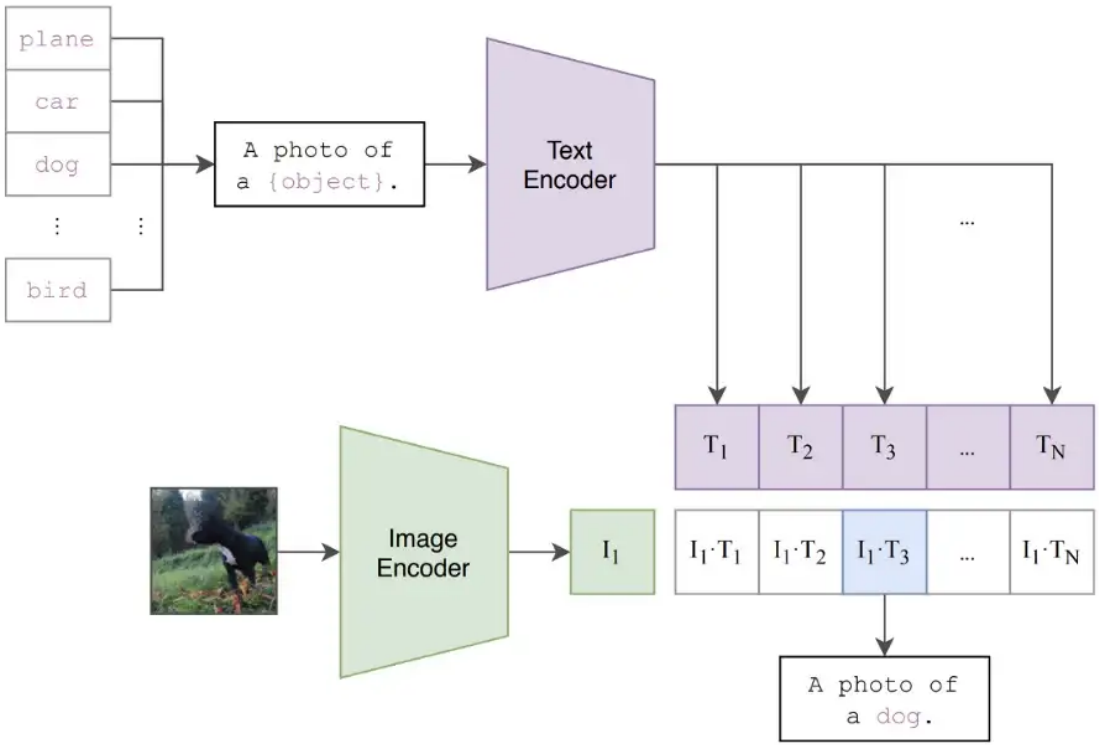
Многоклассовая функция потерь

- Такая цель называется многоклассовая функция потерь N-пар (или InfoNCE)
- Обычно применяется к задачам contrastive и metric learning. В результате этого процесса предварительной подготовки CLIP формирует совместное пространство для эмбедингов изображений и текста, так что изображения и подписи, соответствующие схожим концепциям, имеют аналогичные эмбединги.

Классификация изображения без обучающих примеров

- Как классифицировать изображения без обучающих примеров?
- Способность CLIP выполнять классификацию поначалу может показаться загадкой. Учитывая, что он учится только на неструктурированных текстовых описаниях, как он может обобщать невидимые категории объектов в классификации изображений?
- CLIP обучен предсказывать наличие пары изображения и фрагмента текста. Используя текстовые описания unseen classes (например, имена классов), каждый класс-кандидат может быть оценен путем передачи текста и изображения через соответствующие кодировщики и сравнения полученных эмбедингов.

Сравнение эмбедингов



Формально весь процесс

- Zero-shot learning состоит из следующих шагов:
  - Вычислить эмбединги изображении
  - Вычислить эмбединги для каждого класса из соответствующих текстов (т.е. имен/описаний классов)
  - Вычислите косинусное сходство пар эмбедингов изображений в классе
  - Нормировать по всем близким эмбедингам, чтобы сформировать распределение вероятностей классов

Вопросы

?