Федеральное государственное автономное образовательное учреждение
высшего образования
«Московский физико-технический институт
(национальный исследовательский университет)»
Физтех-школа физики и исследований им. Ландау
Кафедра проблем физики и астрофизики

Направление подготовки / специальность: 03.03.01 Прикладные математика и физика
Направленность (профиль) подготовки: Общая и прикладная физика

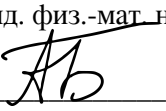
КОНТРАСТНОЕ ОБУЧЕНИЕ В ЗАДАЧАХ КОМПЬЮТЕРНОГО ЗРЕНИЯ ДЛЯ ПОВЫШЕНИЯ ИНТЕРПРЕТИРУЕМОСТИ МОДЕЛИ

(бакалаврская работа)

Студент:
Семёнов Андрей

_____
(подпись студента)

Научный руководитель:
Безносиков Александр Николаевич,
канд. физ.-мат. наук



_____
(подпись научного руководителя)

Консультант *(при наличии)*:
_____
_____
(подпись консультанта)

Москва 2024

Аннотация

Мы предлагаем новую архитектуру и метод объяснимой классификации с использованием концептуальных боттлнек моделей (CBM). В то время как SOTA подходы к задаче классификации изображений работают как черный ящик, растет спрос на модели, которые могли бы предоставлять интерпретируемые результаты. Такие модели часто учатся предсказывать распределение по меткам классов, используя дополнительное описание этих целевых экземпляров, называемое концепциями. Однако существующие методы определения боттлнеков имеют ряд ограничений: их точность ниже, чем у стандартной модели, и CBM требуют дополнительного набора концепций для использования. Мы предоставляем основу для создания концептуальной боттлнек модели на основе предварительно обученного мультимодального энкодера и новых CLIP-подобных архитектур. Представляя новый тип слоев, известный как концептуальные боттлнек слой, мы описываем три метода их обучения: с использованием ℓ_1-потерь, контрастивных потерь и функции потерь, основанной на распределении Gumbel-Softmax (Sparse-CBM), в то время как конечный слой FC по-прежнему обучается с использованием перекрестной энтропии. Мы демонстрируем значительное повышение точности при использовании разреженных скрытых слоев в CBM на основе CLIP. Это означает, что разреженное представление вектора активации концепций имеет смысл в CBM. Более того, с помощью нашего алгоритма поиска по матрице концепций мы можем улучшить предсказание CLIP в сложных наборах данных без какого-либо дополнительного обучения или файн-тьюнинга. Код доступен по ссылке: `https://github.com/Andron00e/SparseCBM`.

Содержание

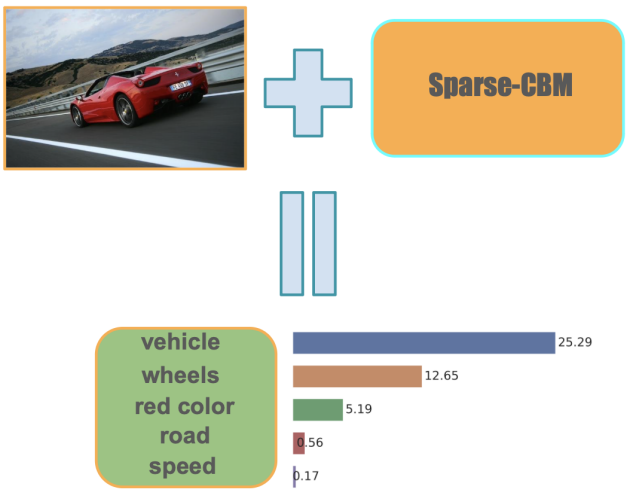


Рис. 1: Пример концептов, отобранных Sparse-CBM.

1 Введение

В последние годы SOTA [1]-подходы к классификации изображений достигли значительных показателей точности. Основным бенчмарком для проверки качества таких моделей является ImageNet LSVRC [2], а наиболее продвинутые модели обучаются от нескольких недель до месяцев на таких масштабных датасетах, как JFT-300M [3] и ImageNet-21K [4] на множестве GPU-машин. Хотя классические современные решения, такие как ViT [5] и другие модели Vision Transformer, превосходят решения на основе ResNet [6], хотя для предварительного обучения требуется значительно меньше вычислительных ресурсов, они по-прежнему работают как черный ящик. Получив на вход мини-пакет изображений и необходимые метки, они выдают готовое распределение по меткам классов. В этом случае целью подхода Concept Bottleneck Models [7] является разработка моделей, которые бы отвечали на такие вопросы, как: "Почему вы выбрали именно этот класс? "На основании каких результатов вы его предсказали?". Предполагается, что этот подход будет чрезвычайно полезен в областях, где объяснения имеют решающее значение, например в медицинских приложениях.

Мы изучаем модели, которые сначала предсказывают промежу-

точный набор понятных человеку понятий, а затем используют его для предсказания конечной метки, поддерживаемой выбранными понятиями. Построение интерпретируемых нейронных сетей стало более популярным с представлением модели OpenAI CLIP [8] и развитием методов контрастивного обучения [9]. Нельзя не отметить растущее направление мультимодальности [10, 11] с его многообещающими результатами по объединению нескольких модальностей, таких как текст, аудио, видео, для построения одной модели, способной решать множество последующих задач. Исследования моделей Concept Bottleneck и Contrastive Learning также подразумевают мультимодальность (между парами изображение-текст) как в контролируемых [12], так и в неконтролируемых [13] условиях, и даже могут пригодиться для приложений Reinforcement Learning [14].

1.1 Контрастное обучение

Наша работа основана в основном на подходах Contrastive Representation Learning к задаче классификации, когда основной целью является изучение пространства вкраплений таким образом, чтобы схожие образцы сближались, а несхожие отдалялись друг от друга. В случае контрастивного обучения мы стремимся выучить совместное латентное пространство вкраплений изображений и текста, как это делают CLIP, ALIGN [15], BLIP [16] и LiT [17]. Контрастные потери на основе softmax [18] стали ключевой задачей предварительного обучения таких моделей. Однако недавняя работа [19] показала возможность превзойти предыдущие результаты с предварительным обучением на основе сигмоидальных потерь, что требует значительно меньше памяти и, таким образом, позволяет обучать модель заблокированного изображения из [17] с гораздо большим размером партии. Большинство методов контрастного обучения применяются для увеличения объема данных, но не только для изображений в компьютерном зрении. Предыдущие работы [20, 21, 22, 23] показали, как текст может быть дополнен без изменения всей семантики предложения.

1.2 Классификация с Concept Bottleneck

Введенные [7], модели узких мест концепций стали популярным направлением в объясняемом ИИ. Общая идея и конвейер обучения узких мест довольно интуитивны: вместо того чтобы решать интересующую задачу напрямую, давайте разделим основную проблему на более мелкие части и извлечем из них наиболее значимые понятия. В качестве концептов можно рассматривать любую понятную человеку информацию. Например, пытаясь классифицировать сломанную руку, хирург-ортопед ищет особые зрительные паттерны, например, трещины на костях, нереальные изгибы рук и т. д. CBM обучаются сначала предсказывать понятные человеку концепции, а затем принимать окончательное решение на основе полученных концепций. Примечательно, что именно для этих целей сквозное обучение не является обязательным, многие предыдущие работы, на которые мы ссылаемся в разделе 2, представляют свои фреймворки для построения CBM на основе существующих предварительно обученных признаков-экстракторов. В чем преимущество CBM, если точность ухудшилась по сравнению с обычным классификатором? Снижение производительности, безусловно, является ключевой проблемой для узких мест в CBM, но вместо этого мы пытаемся использовать этот параметр для определения компромисса между точностью модели и ее интерпретируемостью. Ограничив модель промежуточным набором концепций и конечными предсказаниями слоя с полным подключением (FC), мы можем:

Объяснить, какую информацию модель воспринимает как более/менее важную для классификации входных данных.

Понять, почему модель совершает конкретную ошибку: определить, каким понятиям был отдан неверный приоритет.

За вышеперечисленными преимуществами общих Concept Bottleneck моделей следуют ключевые ограничения:

Подготовка концептов: Для обучения CBM требуются метки, помеченные дополнительными понятиями, сбор которых занимает

много времени и стоит дорого.

Производительность: На практике использование моделей, точность которых ниже, чем у неограниченных, неперспективно.

Редактирование модели: Общая проблема, возникающая в подходе Concept Bottleneck Model, - невозможность целостного вмешательства в саму модель. Предыдущие работы [24] демонстрируют методы обратной связи, управляемой человеком, а [25] полагается на TCAV[26], которая все еще обращает внимание на локальные ошибки модели в случае, когда МД не изучается сквозным образом. В то время как обычно пользователи хотят получить предварительно обученную CLIP-подобную модель и создать над ней "узкое место". Поэтому этот вопрос все еще остается открытым.

1.3 Наш вклад

В этой статье мы вносим вклад в лучшее понимание последних подходов к классификации изображений с помощью CBM. В частности, мы раскрываем возможности CBM, обученных самоконтролем, и обнаруживаем значительный рост производительности моделей, генерирующих разреженное внутреннее представление. Мы стремимся раскрыть ранее скрытые возможности необработанных CLIP подобных моделей с помощью представленного в разделе 4.2 алгоритма "нулевого обучения"с предложенным в разделе 4.3 фреймворком. Только используя оценки точечных продуктов CLIP, мы можем сделать нашу модель более интерпретируемой в смысле опоры на промежуточный набор понятий. Кроме того, все наши подходы поддерживают ручную генерацию концептов. Мы формулируем наш дальнейший фреймворк как частный случай контрастной тонкой настройки, когда предварительно обученные модели затем тонко настраиваются с помощью контрастных объективных функций [27].

Вместо того чтобы разрабатывать архитектуры для конкретных случаев, мы ищем алгоритмы контрастной тонкой настройки, кото-

рые могут работать с общими рамками создания Concept Bottleneck из предварительно обученного мультимодального кодера. Кроме того, наш подход опирается на некоторые ранее представленные схемы с изменением целей, для которых обучаются скрытые слои. Примечательно, что, поскольку структура магистральной модели хорошо распараллеливается, наши CBM можно относительно легко поддерживать вертикальным или информационным параллелизмом. Это позволяет нам спроектировать узкое место, которое может решать свою задачу более эффективно.

Мы резюмируем основной вклад нашей работы следующим образом:

(Вклад 1) Мы формально определяем алгоритм для повышения точности CLIP и в то же время делаем модель более интерпретируемой. Мы также приводим анализ латентного пространства CLIP и рассказываем, как концепции влияют на него.

(Вклад 2) На основе проведенного анализа мы формулируем фреймворк для построения CBM: новую архитектуру и метод обучения, включающий контрастную тонкую настройку дополнительных слоев. Мы реализуем автоматическую генерацию набора концептов на основе предоставленного набора данных и контрастных вариантов наших функций потерь, таким образом, наш фреймворк облегчает обучение новых слоев.

(Вклад 3) Мы продемонстрировали эффективность метода Sparse-CBM (см. раздел 4.5), запустив нашу архитектуру на наборах данных ImageNet [2], CUB-200 [28], Places365 [29], CIFAR100 и CIFAR10 [30]. В результате Sparse-CBM превосходит предыдущую безлаберную [25] работу на нескольких из них и достигает точности 80,02% на CUB-200.

2 Сопутствующие работы

Подходы к классификации изображений с использованием концептуальных моделей - быстро развивающаяся область, поскольку в разделе 1.2 упоминаются преимущества их возможностей для интерпре-

тации, то уже было представлено множество достаточных методов. В ранних работах [7, 31, 32] предлагается обучать CNN [33] и создавать дополнительный слой перед последним полностью связанным слоем, где каждый нейрон соответствует интерпретируемому человеком понятию. Поскольку модели "узкого места"страдают от необходимости подготовки набора концепций, поиск эффективного метода создания набора поддерживаемых концепций является необходимой задачей. В предыдущих работах по безэтикетному, постхоковому CBM и LaBo [25, 24, 34] прослеживается тенденция к созданию фреймворков (как преобразовать существующую модель в модель узкого места) вместо того, чтобы изучать модели с нуля. Они также показывают методы создания набора достаточных концептов и вводят новые метрики в классификацию с помощью CBM. Поскольку хорошо подготовленный набор концептов является ключевым ингредиентом проблемы CBM, [35] предлагает общий обзор методов, использующих вкрапления концептов.Помимо прочего, авторы [25] обучают модель по двум алгоритмам: голова классификатора с решателем GLM-SAGA [36] и узкие слои с вариантами косинусного сходства, что близко к нашему конвейеру. [37] сообщают о методе, который не требует обучения базовой модели, вместо этого они запрашивают у CLIP определенный паттерн и находят конечный класс с помощью предложенной формулы, а [38] предлагает способ описания нейронов скрытых слоев CLIP. [39] предлагает построить входную пирамиду с различными семантическими уровнями для каждой модальности и объединить визуальные и лингвистические элементы в иерархию, что удобно для обнаружения объектов. [36] предлагает идею сохранить конечный слой FC разреженным, что делает его более интерпретируемым. [40] вмешивается в латентное пространство CLIP и показывает, что оно может быть эффективно смоделировано как смесь гауссианов. Также отметим, что в недавней работе [41] предлагается использовать сигмоид Гумбеля после экстракторов признаков (для концептов и изображений), которые дают разреженное представление концептов. Идея разреженности также находит свое продолжение в работе [42], где авторы представляют новый тип слоев Local Winner-Takes-All [43], основанных на активационной разреженности. Предыдущие

работы по разреженности также схожи с нашей, но мы, в свою очередь, создаем варианты контрастных потерь для обучения CBM.

3 Предварительная информация

В этом разделе мы приводим все обозначения, необходимые для представления наших методов обучения моделей Concept Bottleneck.

Нотация. В основном мы работаем с OpenAI CLIP, он состоит из двух кодировщиков, один для изображений, другой для текста. Эти кодировщики позволяют нам получить векторное представление для обоих типов данных в многомерном пространстве одной и той же размерности (обычно 512). Поэтому мы используем следующие обозначения. Мы называем $f_{\mathrm{T}}(t, \theta)$ выходом текстового кодера f_{T}, то есть вкраплениями текста из партии текстовых примеров t. Под θ мы понимаем параметры текстового кодера, если он не поддается обучению, то мы опускаем эти параметры. Также, поскольку в большинстве экспериментов мы настраиваем лишь небольшое количество весов исходной модели CLIP (см. table 2), а полностью обучаем только новые встроенные слои, для простоты мы указываем веса θ с одинаковым символом для обоих кодеров. Получив мини-пакет изображений x, мы используем $f_{\mathrm{I}}(x, \theta)$ для кодировщика изображений f_{I} на выходе, т.е. вкраплений входных изображений, каждое размерностью 512 для основных конфигураций модели CLIP. $\langle , \rangle$ обозначает скалярное (точечное) произведение, через $\times$ мы обозначаем декартово произведение двух множеств. Для векторов $\|z\|$ - это норма на векторном пространстве, которому принадлежит z. Как правило, под этим обозначением подразумевается 2-норма для векторов из $\mathbb{R}^n$ или норма Фробениуса для матриц из $\mathbb{R}^{m \times n}$, если не указано иное.

4 Постановка задачи и основные Результаты

В этом разделе мы представляем наш алгоритм и фреймворк для построения CBM. Алгоритм, представленный в разделе 4.2, обеспечивает повышение точности CLIP наряду с его интерпретируемостью, а фреймворк (см. раздел 4.3) обеспечивает контрастную тонкую настройку магистрального бимодального кодера.

4.1 Постановка задачи

Наряду с задачей классификации изображений, мы формализуем классификацию с помощью **Concept Bottleneck**. Учитывая кодировщики изображений и текста, мы намерены выучить дополнительную проекцию из пространства вложений, предоставляемого кодировщиками, в векторное пространство, соответствующее представлению концептов изображений, а затем выучить окончательную проекцию для получения вероятностей меток классов. Мы намерены выучить следующее отображение: $x \times t \to \omega \to l$, согласно *Notation* 3: x обозначает область изображений, t - текстовые понятия, а ω и l - представление понятий и метки классов соответственно. Это отличается от стандартной схемы $x \to \omega \to l$ тем, что мы можем создавать концепты t автоматизированным способом.

4.2 Concept Matrix Search algorithm

Мы представляем алгоритм Concept Matrix Search (CMS), который использует возможности CLIP для представления изображений и текстов в совместном латентном пространстве одной и той же размерности. Сначала мы формулируем гипотезу о наших данных, принимая во внимание то, как обучена основная модель CLIP.

Определим набор эмбеддингов картинки как $\mathrm{I} = \{i_1, \ldots, i_{|\mathrm{I}|}\}$, where $i_\mathrm{k} = f_\mathrm{I}(x_\mathrm{k}, \theta)$. И в качестве $\mathrm{D} = \{d_1, \ldots, d_{|\mathrm{D}|}\}$ мы определим набор эмбеддингов концептов, $d_\mathrm{j} = f_\mathrm{T}(\mathrm{concept}_\mathrm{j}, \theta)$, тем временем

$\mathrm{C} = \{c_1, \ldots, c_{|\mathrm{C}|}\}$ набор эмбеддингов классов, где эмбеддинги меток этих классов похоже выражены ів терминах f_{T}. Мы также определим две матрицы для которых наша модель будет вычислять попарную близость картинка-текст:

Image-Concept matrix (V-matrix): $\mathcal{V} \in \mathbb{R}^{|\mathrm{I}|\times|\mathrm{D}|}$, such that $\mathcal{V}_{\mathrm{kl}} = \langle i_{\mathrm{k}}, d_{\mathrm{l}} \rangle$.

Class-Concept matrix (T-matrix): $\mathcal{T} \in \mathbb{R}^{|\mathrm{C}|\times|\mathrm{D}|}$, such that $\mathcal{T}_{\mathrm{kl}} = \langle c_{\mathrm{k}}, d_{\mathrm{l}} \rangle$.

Теперь мы можем сформулировать *гипотезу* лежащую в основании CMS.

Гипотеза. Для $f_{\mathrm{I}}(x, \theta)$ and $f_{\mathrm{T}}(t, \theta)$ определим косинусную близость как:

$$\cos(f_{\mathrm{I}}(x, \theta), f_{\mathrm{T}}(t, \theta)) = \frac{\langle f_{\mathrm{I}}(x, \theta), f_{\mathrm{T}}(t, \theta) \rangle}{\|f_{\mathrm{I}}(x, \theta)\| \|f_{\mathrm{T}}(t, \theta)\|}.$$

Тогда,

> *Для матриц $\mathcal{V}$, $\mathcal{T}$ и функции $\phi(\mathrm{m})\colon \mathbb{N}^{|\mathrm{I}|} \to \mathbb{N}^{|\mathrm{C}|}$ которая отображает индекс эмбеддинга картинки $\mathcal{V}$ в индекс его класса из $\mathcal{T}$, мы предполагаем:*
>
> $$\forall \mathrm{j} \neq \mathrm{k} \hookrightarrow \cos(\mathcal{V}^{\top}_{\mathrm{k},\cdot}, \mathcal{T}^{\top}_{\phi(\mathrm{k}),\cdot}) \geq \cos(\mathcal{V}^{\top}_{\mathrm{k},\cdot}, \mathcal{T}^{\top}_{\phi(\mathrm{j}),\cdot}),$$
>
> *где $\mathcal{V}_{\mathrm{k},\cdot}$, $\mathcal{T}_{\mathrm{k},\cdot}$ k-ые строки $\mathcal{V}$, $\mathcal{T}$ соответственно.*

Проще говоря, *гипотеза* гласит, что для всех возможных классов вектор сходства между истинным классом и всеми концептами должен быть наиболее близок к вектору сходства образа этого самого класса и всех концептов. Вектор $\mathcal{V}^{\top}_{\mathrm{k},\cdot}$ получается путем вычисления косинуса сходства между эмбеддингами i_{k} и каждым из вложений концепта в D.

CMS алгоритм. Сформулировав *гипотезу*, мы предлагаем эффективный метод ее проверки с помощью algorithm 1. Для сокращения вычислительных затрат и памяти мы предлагаем в algorithm 1

Algorithm 1 Concept Matrix Search

Input: Batch of image embeddings $\mathrm{I}_{|\mathrm{B}|}$, labels, all classes C and concepts D embeddings.
Build $\mathcal{V} \in \mathbb{R}^{|\mathrm{B}|\times|\mathrm{D}|}$, $\mathcal{T} \in \mathbb{R}^{|\mathrm{C}|\times|\mathrm{D}|}$ matrices, store $\mathcal{T}$.
for $\mathrm{k} = 0, 1, 2, \ldots, |\mathrm{B}| - 1$ **do**
    for $\mathrm{m} = 0, 1, 2, \ldots, |\mathrm{C}| - 1$ **do**
        Compute and store $\cos(\mathcal{V}_{\mathrm{k},\cdot}^{\top}, \mathcal{T}_{\mathrm{m},\cdot}^{\top})$
    end for
    Find $\mathrm{m}_{\max} = \max_{\mathrm{m}} \cos(\mathcal{V}_{\mathrm{k},\cdot}^{\top}, \mathcal{T}_{\mathrm{m},\cdot}^{\top})$
    if $\mathrm{label}(\mathrm{k}) = \mathrm{m}_{\max}$ **then**
        the hypothesis has been proven, increase Accuracy
    else
        the hypothesis has been disproved
    end if
end for
return Mean accuracy

обрабатывать матрицу $\mathcal{V}$ партиями с размером партии $|\mathrm{B}| < |\mathrm{I}|$. Обратите внимание, что на самом деле мы не выполняем два цикла, а используем эффективное умножение матриц, реализованное в PyTorch [44].

Предлагаемый алгоритм обеспечивает повышение интерпретируемости исходной модели CLIP, не требуя дополнительного обучения. Простую демонстрацию нашего метода можно увидеть на рисунке fig. 10. В этой работе мы также обнаружили влияние набора концептов на точность CMS, которое представлено на рисунке 5 и обсуждается в экспериментах (см. раздел 5.2). Мы также представляем эмпирическое подтверждение гипотезы CMS *гипотеза*, рассматривая точность классификации каждого класса отдельно, и показываем явную зависимость от набора концепций [1].

[1] https://github.com/anonymousedauthor/SparseCBM/blob/main/additional_evaluations/cms_empirical_evidence.ipynb

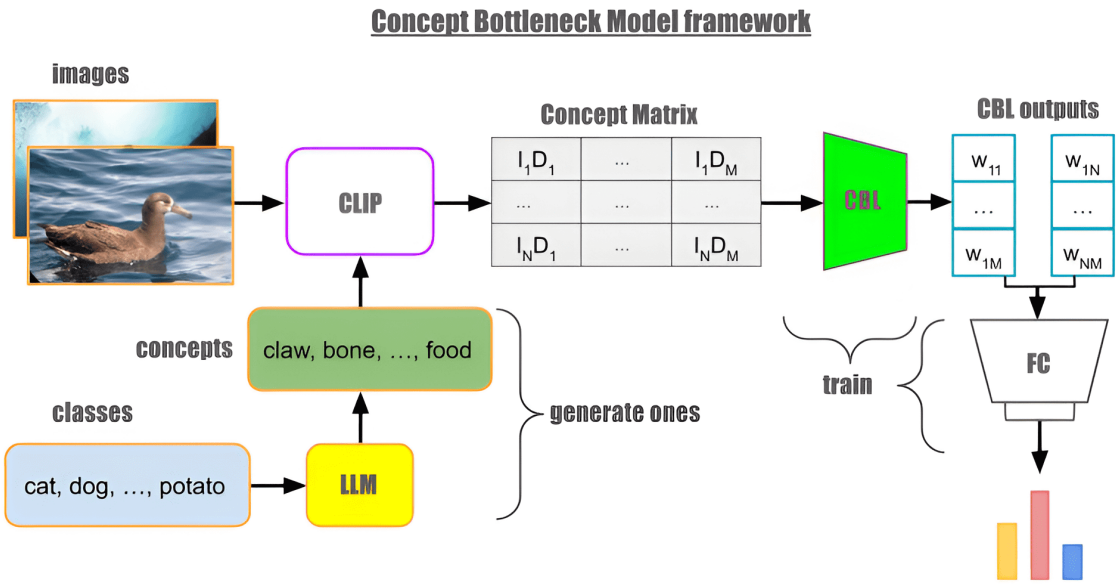


Рис. 2: Архитектура фреймворка для построения CBM.

4.3 Наш фреймворк для CBMs

В этом разделе мы предлагаем структуру, которая создает Concept Bottleneck Model из предварительно обученного мультимодального кодера эффективным способом с минимальным количеством настраиваемых слоев и автоматической генерацией концептов для каждого рассматриваемого набора данных с точной настройкой. Сначала наш МД строит модель, подходящую для классификации изображений, а в качестве основы мы используем бимодальный кодер или CLIP-подобные модели, способные выдавать вкрапления или готовые к использованию оценки точечного произведения этих вкраплений. Это означает, что последний слой сети модифицируется в зависимости от количества классов в наборе данных. Во-вторых, количество концептов и их значение зависят только от меток набора данных и не меняются в процессе дальнейшего обучения. Мы можем кратко описать строительные блоки нашего фреймворка в виде следующих шагов:

- **Шаг 1:** Выберите подходящую мультимодальную модель, основанную на кодировании, в качестве основы (мы в основном используем OpenAI CLIP-ViT-L/14[2]). Затем добавляем к ней два линейных слоя, первый из которых мы называем Concept Bottleneck Layer (CBL), как в [25], а второй - последний FC-слой.

- **Шаг 2:** Выбираем набор данных, создаем набор концептов на основе меток классов и фильтруем[3] нежелательных концептах.

- **Шаг 3:** Выбираем одну из наших целевых функций, упомянутых в разделах 4.5, 4.6, 4.4 и два подходящих оптимизатора для полученной архитектуры (мы используем Adam [48] и AdamW [49]).

- **Шаг 4:** Выучите CBL с подобранной на предыдущем шаге целью и FC с кросс-энтропийным лоссом.

Более конкретно. На **шаге 1** мы рассматриваем CLIP-подобную модель и добавляем к ней два линейных слоя. Для заданной партии входных изображений с **всеми** созданными концептами эта модель выводит оценки точечного произведения между конкретным изображением и всеми концептами, то есть, в терминах алгоритма CMS (раздел 4.2), мы получаем партию размером $|\mathrm{B}|$ строк матрицы $\mathcal{V}$. Эти оценки показывают, насколько чувствительны изображения от 1 до $|\mathrm{B}|$ к каждой концепции, поскольку более подходящие концепции получают более высокую оценку от CLIP. Первый линейный слой, назовем его CBL, управляет информативностью концептов, а второй FC делает окончательное предсказание. Мы храним CBL и FC как матрицы формы $|\mathrm{D}| \times |\mathrm{D}|$ и $|\mathrm{D}| \times |\mathrm{C}|$ соответственно. Единственное концептуальное различие между этими слоями заключается в том, как они обучаются. На **шаге 2** мы генерируем набор концептов. Как

[2]В экспериментах мы используем модель Hugging Face [45], представленную на сайте `https://huggingface.co/openai/clip-vit-large-patch14`.

[3]В экспериментах мы используем 'all-mpnet-base-v2' трансформатор предложений [46] , присутствующий в `https://huggingface.co/sentence-transformers/all-mpnet-base-v2`.

и в [25], мы не полагаемся ни на каких экспертов и создаем понятия вручную. Один из распространенных способов - создавать их, запрашивая у LLM метки классов и обрабатывая результаты работы LLM. Мы спрашиваем GPT-3 [50], Llama-2[4]. Обе модели демонстрируют хорошее знание того, какие концепции более предпочтительны для каждого класса. Мы напрямую задаем следующий вопрос:

Prompt 1: "List the most important features for recognizing something as a {label}. Write them one by one."

Prompt 2: "List the things most commonly seen around a {label}. Write them one by one."

Prompt 3: "Give a generalization for the word {label}. Answer with a single sentence."

Также, в связи с правилами оплаты OpenAI API и размером моделей Llama-2, мы используем ConceptNet API [52] для создания концептов также в автоматическом режиме, но более просто и бесплатно, как в [25, 24]. При использовании ConceptNet у нас нет возможности подсказывать, вместо этого мы используем Sentence Transformer [46] и выбираем дальнейшие понятия с помощью алгоритма работы, аналогичного [25]:

1) Чтобы представить понятия в виде нескольких слов, мы удаляем все понятия, длина которых превышает 30 букв.

2) Мы удаляем все понятия, у которых косинусоидальное сходство с классами больше 0,85.

3) Мы удаляем все понятия, у которых косинусоидальное сходство с другими понятиями больше 0,9 отсечки[5].

4) Мы удаляем понятия с меньшим средним сходством с другими понятиями, чтобы оставить более общие понятия.

На **Шаг 4** мы, в свою очередь, предлагаем применить *последовательное узкое место* схемы: пусть мы обучаем эти слои одновременно, но применяем оптимизатор FC ко всем обучаемым параметрам

[4]В экспериментах мы используем бесплатную версию Llama-2, доступную на Hugging Face hub `https://huggingface.co/TheBloke/Llama-2-70B-Chat-GPTQ`. [51] для этих целей

[5]0,85 для сходства "понятия-классы"и 0,9 для сходства "понятия-концепты это наши гиперпараметры, полученные эмпирическим путем, мы не собираемся менять их в дальнейших экспериментах

сети, кроме CBL, который обучается только со своим собственным оптимизатором. Если CLIP выдает $\psi(x,t) = \big(\langle i, d_1\rangle, \ldots, \langle i, d_{|\mathrm{D}|}\rangle\big)^\top \ \mathbb{R}^{|\mathrm{D}|}$ с вложением изображения i и вложением понятия d_j из набора данных $\mathcal{D} = (x, t, l)$, где x обозначает визуальную модальность, t обозначает **all** слова понятия, а l - текстовое описание имени класса; функция потерь для CBL $\mathcal{L}_{\mathrm{CBL}}$; сама CBL $W_{\mathrm{CBL}} \in \mathbb{R}^{|\mathrm{D}|\times|\mathrm{D}|}$; и потери кросс-энтропии финального слоя и веса $\mathcal{L}_{\mathrm{CE}}$, $W_{\mathrm{F}} \in \mathbb{R}^{|\mathrm{C}|\times|\mathrm{D}|}$. Мы можем сформулировать период обучения для CBL следующим образом:

$$\min_{W_{\mathrm{CBL}}} \mathop{\mathbb{E}}_{(x,t,l)\sim\mathcal{D}} \big[\mathcal{L}_{\mathrm{CBL}}(W_{\mathrm{CBL}}\psi(x,t))\big],$$

и, на предыдущем шаге бэкварда, мы учим FC слой с

$$\min_{W_{\mathrm{F}}} \mathop{\mathbb{E}}_{(x,t,l)\sim\mathcal{D}} \big[\mathcal{L}_{\mathrm{CE}}(W_{\mathrm{F}}W_{\mathrm{CBL}}\psi(x,t), l)\big].$$

Таким образом, мы обучаем CBL и FC одновременно, но обновления градиента от Cross-Entropy loss не влияют на W_{CBL}, что очень удобно при обучении слоев узких мест с ℓ_1 или функций потерь Gumbel-Softmax, чтобы сделать эти слои разреженными. Здесь мы показываем, что $\mathcal{L}_{\mathrm{CE}}$ ожидает двух аргументов в качестве типичной супервизорной потери, в то время как $\mathcal{L}_{\mathrm{CBL}}$ требует только одного аргумента в качестве самоподдерживающейся функции потерь. Для обоих уравнений мы используем оптимизаторы Adam или AdamW, подключенные к соответствующим слоям. Ключевое преимущество нашего фреймворка заключается в том, как обучить эту МД с двумя оптимизаторами: по одному для каждой из представленных выше задач; и какие функции потерь выбрать (см. разделы 4.4, 4.5). А для случая обучения с потерями ℓ_1 мы рассматриваем термин регуляризации, введенный в разделе 4.6.

В fig. 2 мы приводим схему нашего общего фреймворка в виде архитектуры с настроенными слоями и генерацией концепций.

4.4 Contrastive-CBM

В этом разделе мы предлагаем простую адаптацию потери CLIP для обучения концептуальных слоев. Использование контрастирующей цели вместо попытки предсказать точные слова, связанные с изображениями, - это то, что делает модель на основе CLIP популярной для классификации изображений с нулевого снимка и сходства изображений и текстов. Когда мы создаем МД на основе предварительно обученного мультимодального кодера и снабжаем его необходимым набором концептов, мы также можем свободно обучать его узкие слои с теми же контрастными потерями [19].

Но в нашем случае мы заставляем CBL обучаться, минимизируя не сходство CLIP "изображение-текст а выходы слоя узкого места $W_{\mathrm{CBL}}\psi(x,t)$ с учетом параметров W_{CBL}. Пусть $|\mathrm{B}|$ - размер партии, в которой задана мини-пария вкраплений $\mathrm{B} = \big((i_1, d_1, c_1), \ldots, (i_{|\mathrm{B}|}, d_{|\mathrm{B}|}, c_{|\mathrm{B}|})\big)$. Для простоты мы приводим формулу для случая, когда количество концептов равно размеру партии $|\mathrm{D}| = |\mathrm{B}|$. Мы также обозначим $\varphi_{\mathrm{k}} \triangleq \big(\langle i_{\mathrm{k}}, d_1\rangle, \ldots, \langle i_{\mathrm{k}}, d_{|\mathrm{B}|}\rangle\big)^\top$ как вектор оценок точечного произведения между **all** концептами ($|\mathrm{D}| = |\mathrm{B}|$) и k-ым изображением из партии B. В качестве w_{k} мы определяем k-ю строку матрицы W_{CBL}. Тогда наши контрастные потери для обучения узкого слоя можно переписать следующим образом:

$$-\frac{1}{2|\mathrm{B}|}\sum_{\mathrm{k}=1}^{|\mathrm{B}|}\left(\log\frac{e^{\alpha\langle w_{\mathrm{k}},\varphi_{\mathrm{k}}\rangle}}{\sum_{\mathrm{j}=1}^{|\mathrm{B}|}e^{\alpha\langle w_{\mathrm{k}},\varphi_{\mathrm{j}}\rangle}}+\log\frac{e^{\alpha\langle w_{\mathrm{k}},\varphi_{\mathrm{k}}\rangle}}{\sum_{\mathrm{j}=1}^{|\mathrm{B}|}e^{\alpha\langle w_{\mathrm{j}},\varphi_{\mathrm{k}}\rangle}}\right).$$

Скаляр α параметризуется как $\exp\tilde{\alpha}$[6]. Мы определяем "Contrastive-CBM"как частный случай концептуальной модели узких мест нашего фреймворка с контрастной целью $\mathcal{L}_{\mathrm{CBL}}$ для слоев узких мест в представленной выше форме. Мы также отмечаем, что этот вид потерь может быть эффективно реализован для распределенного обучения [53, 19].

[6]В OpenAI CLIP $\tilde{\alpha} = 1.155$, который мы используем для CMS backbone CLIP, в то время как для CBM используется значение 2.659

4.5 Sparse-CBM

В этом разделе мы предлагаем основную целевую функцию для обучения наших моделей. Во-первых, мы определяем распределение Gumbel-Softmax [54, 55] следующим образом: пусть z - категориальная переменная с вероятностями $\pi_1, \cdots, \pi_{|\mathrm{D}|}$, то есть, $|\mathrm{D}|$-мерный однохотовый вектор из вероятностного симплекса $\Delta^{|\mathrm{D}|-1} \triangleq \{\pi \in \mathbb{R}_+^{|\mathrm{D}|} : \sum_{i=1}^{|\mathrm{D}|} \pi_i = 1\}$. Тогда трюк Gumbel-max [56, 57] позволяет нам сделать выборку z из категориального распределения с вероятностями классов $\pi = (\pi_1, \cdots, \pi_{|\mathrm{D}|})$:

$$z = \mathbf{1}\left(\arg\max_{\mathrm{k}} \left[g_{\mathrm{k}} + \log \pi_{\mathrm{k}}\right]\right),$$

где $g_1, \ldots, g_{|\mathrm{D}|}$ - i.i.d. выборки из $\mathrm{Gumbel}(0, 1)$, которые, в свою очередь, могут быть получены из $u\ in\mathrm{Uniform}(0, 1)$ путем выборки $g = -\log\log u$, (обозначим индикатор $\mathbf{1}$ как функцию одной точки). После применения функции softmax как непрерывной дифференцируемой аппроксимации к $\arg\max$ мы получаем распределение Гумбеля-Softmax, непрерывное распределение на симплексе, которое может аппроксимировать выборки из категориального распределения. Затем мы намерены построить контрастные потери для CBL-выводов следующим образом:

$$-\frac{1}{2|\mathrm{B}|}\sum_{\mathrm{k}=1}^{|\mathrm{B}|}\left(\log\frac{e^{(\log(\alpha\langle w_{\mathrm{k}},\varphi_{\mathrm{k}}\rangle)+g_{\mathrm{k}})/\tau}}{\sum_{\mathrm{j}=1}^{|\mathrm{B}|} e^{(\log(\alpha\langle w_{\mathrm{k}},\varphi_{\mathrm{j}}\rangle)+g_{\mathrm{j}})/\tau}} + \log\frac{e^{(\log(\alpha\langle w_{\mathrm{k}},\varphi_{\mathrm{k}}\rangle)+g_{\mathrm{k}})/\tau}}{\sum_{\mathrm{j}=1}^{|\mathrm{B}|} e^{(\log(\alpha\langle w_{\mathrm{j}},\varphi_{\mathrm{k}}\rangle)+g_{\mathrm{j}})/\tau}}\right).$$

Мы используем полученную потерю для представления логитов слоя "узкого места"как категориальных переменных, т.е. разреженных векторов, которые хорошо интерпретируются [36]. С помощью структуры Гумбеля-Софтмакса $\mathcal{L}_{\mathrm{CBL}}$ достигается разреженное обучение слоев узкого места, что повышает их интерпретируемость: ключевым ингредиентом для обеспечения разреженности выходных векторов CBL является температура τ. Представленное распределение

является гладким для $\tau > 0$, оно имеет определенные градиенты по параметрам w_{k}. Заменив категориальные выборки на выборки Гумбеля-Софтмакса, мы можем осуществлять обратное распространение через слои Concept Bottleneck. Для низких температур ($\tau < 0,5$) ожидаемое значение распределения Гумбеля-Софтмакса приближается к ожидаемому значению категориальной случайной величины с теми же логитами, т.е. становится одномоментным. С ростом температуры ожидаемое значение сходится к равномерному распределению по категориям. На практике мы начинаем с высокой температуры и отжигаем до небольшой, но ненулевой температуры.

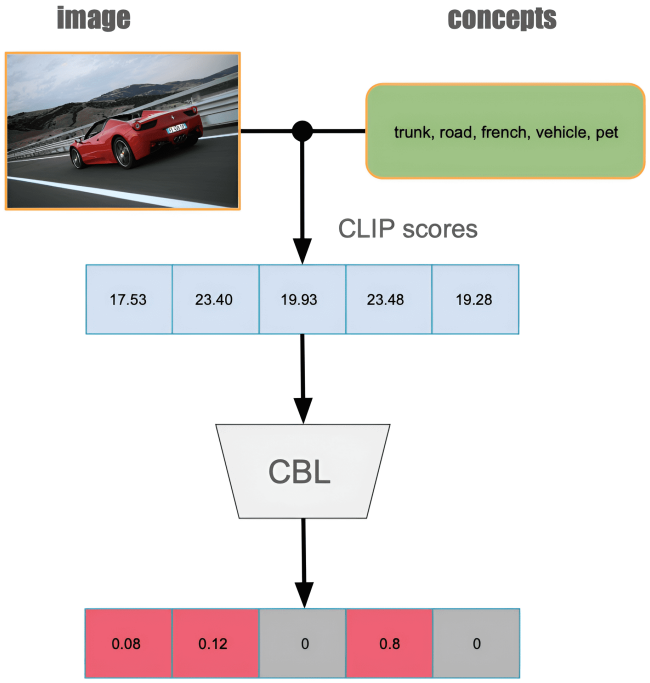


Рис. 3: Визуализация Sparse-CBM Concept Bottleneck слоев.

Если τ является обучаемым параметром (а не отжигается по фиксированному расписанию), эту схему можно интерпретировать как энтропийную регуляризацию [58, 59]. Высокоуровневое объяснение и вывод активаций Гумбеля можно посмотреть на [60].

4.6 ℓ_1-CBM

Мы также обучаем концепт Bottleneck с помощью ℓ_1-функции потерь. Ранее [24] предлагал использовать штраф эластичной сети, а [25] - среднее взвешенное между нормой Фробениуса и нормой матрицы элементов. Вместо этого мы показываем многообещающий результат только при использовании штрафа ℓ_1. Штраф ℓ_1 демонстрирует способность к спарсированию обучаемых слоев: оптимизатор пытается минимизировать потери, уменьшая веса, которые имеют ненулевой градиент, что приводит к тому, что некоторые веса устанавливаются точно в ноль, что эффективно удаляет соответствующие признаки из выходных данных CBL.

Начнем с определения следующей оптимизационной задачи:

$$\min_{W_{\text{CBL}}} \mathop{\mathbb{E}}_{(x,t,l)\sim\mathcal{D}} \left[\mathcal{L}_{\text{CE}}(W_{\text{F}} W_{\text{CBL}} \psi(x,t), l) + \frac{\lambda}{|\text{D}|}\Omega(W_{\text{CBL}})\right],$$

где $\Omega(W_{\text{CBL}})$ соответсвует регуляризатору. Мы используем:

$$\Omega(W_{\text{CBL}}) = \|W_{\text{CBL}}\|_1$$

с единтвенной параметризацией λ.

5 Эксперименты

Бейслайн. Здесь мы приводим основные сведения о моделях и алгоритмах, с которыми мы сравниваем нашу.

1) Мы оцениваем эффективность описанного фреймворка (раздел **??**), в частности, Sparse-, Contrastive-, ℓ_1-CBMs, сравнивая его с предыдущими Label-free [25], Post-hoc CBM [24], LaBo [34] фреймворки на одних и тех же наборах данных [4, 28, 30, 29], но с разными базовыми моделями: в то время как в экспериментах с фреймворком Label-free, проведенных с ResNet50 [6] и CLIP(ResNet50)[8], мы используем только варианты архитектуры CLIP в качестве основы. Заметим, что оба фреймворка получают довольно схожие концепции, поскольку Label-free также включает опцию генерации концепций с

помощью ConceptNet API. Также модель Label-free обучается двумя алгоритмами последовательно: сначала обучается CBLs с cos-cubed сходством, введенным в [25], затем последний FC-слой обучается с помощью решателя GLM-SAGA [36]. Мы также приводим сравнение с линейным зондированием, когда один линейный слой после CLIP обучается классификации. Подробное объяснение эксперимента с зондированием мы приводим в appendix A.

2) Мы оцениваем эффективность описанного алгоритма Concept Matrix Search (раздел 4.2), сравнивая его с предыдущей работой Visual Classification via Description, которую мы обозначаем как метод "DescriptionCLS"в table 2. Оба метода схожи в том, что они способны различать классы, выбирая тот, который имеет наивысший балл, предоставляемый некоторым *rule*, и оба они не обучают никакую модель и используют только эмпирические формулы для предсказания целей более интерпретируемым способом. Наш *rule* - это CMS (см. algorithm 1), в то время как метод "DescriptionCLS"опирается на функцию, которая дает оценку классу на основе того, сколько концептов этого класса чувствительны к изображению этого самого класса. Чувствительность понятия измеряется как логарифмическая вероятность того, что понятие относится к изображению в соответствии с косинусным сходством. CMS также использует эту меру, но она не создает прямого отображения класс-концепт, вместо этого мы проводим классификацию в более общем виде, учитывая только партию изображений и все концепции с классами. Кроме того, Concept Matrix Search играет роль более общего классификатора, и мы сравним их ниже, что весьма информативно, поскольку оба метода используют одну и ту же модель CLIP в качестве основы. Кроме того, необходимо привести сравнение с классификацией изображений с нулевым снимком. В этом случае мы рассматриваем пакет изображений и названия классов как входные данные для кодирования CLIP, который, в свою очередь, выдает матрицу "изображение - класс". Выполнив операцию argmax, мы определяем наиболее близкий класс.

Таблица 2: Сравнение перформанса Bottleneck моделей на основных датасетах. Мы наблюдаем превосходство Sparse-CBM над другими архитектурами на CIFAR10, CIFAR100 и CUB200 датасетах.

Model	CIFAR10	CIFAR100	ImageNet	CUB200	Places365
Sparse-CBM (Ours)	**91.17%**	**74.88%**	71.61%	**80.02%**	41.34%
ℓ_1-CBM (Ours)	85.11%	73.24%	71.02%	74.91%	40.87%
Contrastive-CBM (Ours)	84.75%	68.46%	70.22%	67.04%	40.22%
Label-free CBM	86.40%	65.13%	**71.95%**	74.31%	**43.68%**
Post-hoc CBM (CLIP)	83.34%	57.20%	62.57%	63.92%	39.66%
LaBo (full-supervised)	87.90%	69.10%	70.40%	71.80%	39.43%
Linear Probing	96.12%	80.03%	83.90%	79.29%	48.33%

Таблица 3: Сравнение CMS и "DescriptionCLS"на основных датасетах.

Method	CIFAR10	CIFAR100	ImageNet	CUB200	Places365
CMS (Ours)	**85.03%**	62.95%	**77.82%**	**65.17%**	39.43%
DescriptionCLS	81.61%	**68.32%**	75.00%	63.46%	40.55%
Zero-shot	81.79%	52.84%	76.20%	62.63%	**41.12%**

5.1 CBM эксперименты

Набор концептов. Мы сгенерировали большие наборы: один с 6 500 концептами, используя Llama-2, и второй с 5 000 концептами, используя ConceptNet [52] API, и обработали их с помощью трансформатора предложений 'all-mpnet-base-v2', чтобы найти отсечения сходства, упомянутые в **шаг 2** (см. section 4.3). API и обработали их с помощью трансформатора предложений 'all-mpnet-base-v2', чтобы найти отсечки сходства, упомянутые в **Шаг 2** (см. section 4.3). Для самого большого набора мы подготовили все метки классов, чтобы создать из них концепты. Более того, для каждого набора данных

Таблица 1: Зависимость размера набора концептов от набора данных.

Dataset	# of concepts
CIFAR10	120
CIFAR100	944
ImageNet	4,751
CUB200	926
Places365	2,900
"All concepts"	5,051

в table 1 также представлен соответствующий набор концептов. Мы называем самый большой набор концептов "Все концепты который можно увидеть в fig. 5. Каждый набор данных имеет свой собственный набор концептов, сгенерированный из классов набора данных в соответствии с **шаг 2**. Для корректного сравнения, представленные в table 2 модели были обучены *only* с концептами, сгенерированными ConceptNet.

Модели. CLIP-ViT-L/14 и B/32[7] в качестве основы для всех CBM-экспериментов. В зависимости от набора данных, размер этих моделей варьируется. Самая маленькая конфигурация B/32 для CIFAR10 имеет размер 151,3 млн параметров и 50 000 обучаемых параметров, что эквивалентно 0,6 ГБ. Самая большая - L/14 для ImageNet-1K: 455 миллионов параметров; 27,3 миллиона обучаемых параметров и размер 1,81 ГБ. Для наглядности мы показываем точную информацию в table 2 (до обучения). А в table 2 мы приводим результаты для конфигурации CLIP-ViT-L/14.

Оборудование. Мы обучали наши модели на одной машине с 4 графическими процессорами NVIDIA A100-SXM4-80GB и A100-PCIE-40GB, по два каждого типа. Для наших конфигураций МД каждый

[7]Упомянутую модель CLIP-ViT-B/32 можно найти на хабе Hugging Face `https://huggingface.co/openai/clip-vit-base-patch32`

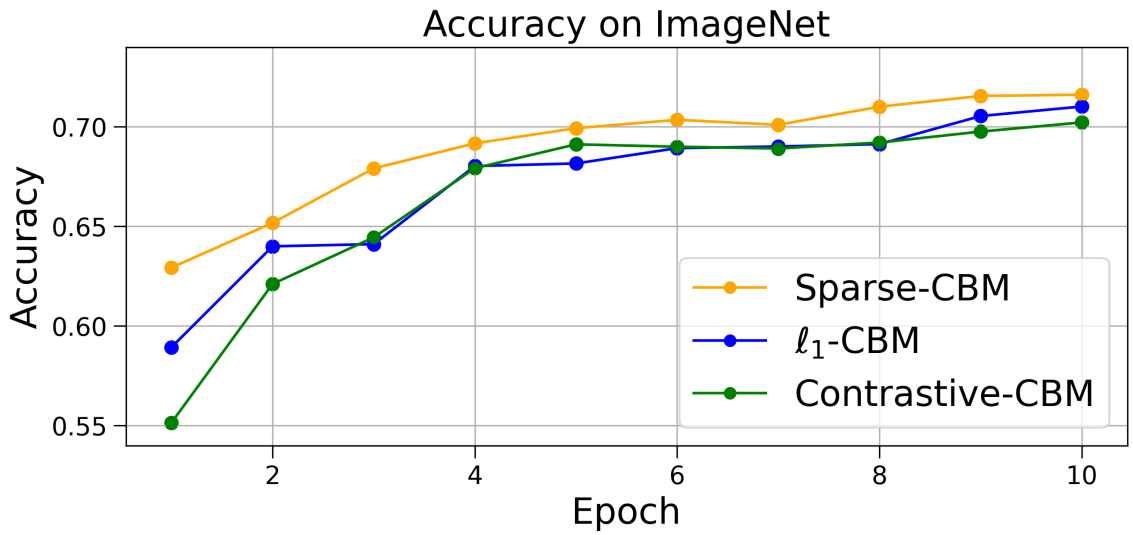


Рис. 4: Сравнение CBM мотодов на валидационной подвыборке датасета ImageNet.

шаг обучения со скоростью обучения по умолчанию, указанной в таблицах 2, 1 данных, занял менее 1 секунды для конфигурации CLIP-ViT-B/32 backbone и немного больше для CLIP-ViT-L/14. Мы обучили каждую из моделей Sparse, ℓ_1, Contrastive-CBM на ImageNet-1K [2] за 20 000 шагов, что заняло чуть больше 5,5 часов для каждой модели (см. fig. 9). Наши реализации не зависят от конкретных аппаратных архитектур.

Рабочая нагрузка . Мы поддерживаем схему *последовательных узких мест* из [7]. Действительно, сначала мы производим градиентное обновление весов W_{F} и W_{CBL} с учетом их параметров независимо друг от друга, а затем, во время следующего прохода вперед, FC получает на вход логиты из обновленного CBL. Таким образом, после каждой итерации задач минимизации CBL и FC для классификатора создается новое представление концепта.

5.2 CMS эксперименты

В этом разделе мы сравним два алгоритма: наш CMS 1 и основной метод, предложенный в [37]. Оба они используют базовые возможности модели CLIP. Мы тестируем их на одной и той же конфигурации CLIP-ViT-L/14. Мы показали превосходство нашего метода на нескольких наборах данных, которые можно увидеть в table 3. Авторы [40] измерили зависимость точности при обнулении определенного количества концептов, а [61] исследовали влияние нескольких относительных концептов. Мы же измерили зависимость точности от различных наборов понятий: почти синонимов (сходство "понятие-класс"0,95 вместо 0,85), сгенерированных ConceptNet как обычно, одного набора многих понятий (большинство из них бесполезны для данных CIFAR10) и 3 000 случайно сгенерированных слов. Результаты представлены в fig. 5.

6 Обсуждение

table 2 демонстрирует превосходство Sparse-CBM по сравнению с Label-free на всех наборах данных, кроме ImageNet и Places365. Более того, Contrastive-CBM имеет самый низкий общий балл. Мы интерпретируем эти наблюдения следующим образом:

1) Из-за table 1 пропорция между размером CBL и размером FC различается. В то время как такие наборы данных, как CIFAR10 и CIFAR100, содержат в ≈ 10 раз больше концептов, чем классов, для ImageNet и Places365 это число, а значит и пропорция между размерами слоев, примерно в 4,6-4,7 раза больше. Мы настаиваем на том, что разреженность внутренних слоев приносит больше пользы при увеличении размерности CBL по сравнению с размерностью выхода последнего полностью связанного слоя.

2) Методу Contrastive-CBM не хватает интерпретируемости представления концептов, т. е. разреженности CBL (подробное сравнение интерпретируемости см. в appendix D.2).

Мы также ссылаемся на алгоритм CMS и показываем снижение

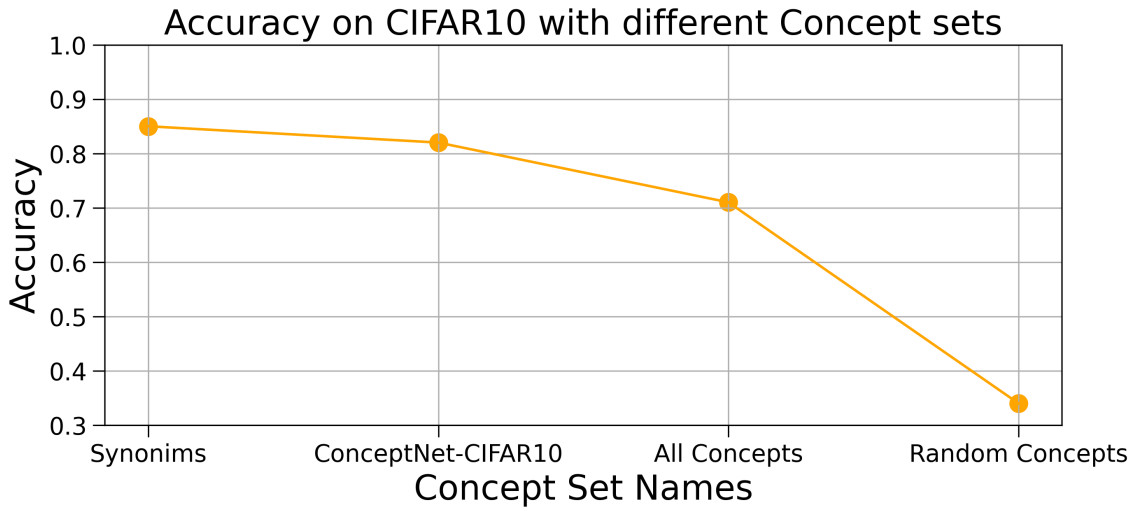


Рис. 5: Зависимость качества CMS от набора конйептов.

точности в зависимости от "качества"концептов, что делает наш метод и подобные ему менее универсальными. С другой стороны, при "хорошем"наборе понятий для каждого набора данных мы показываем, что CMS превосходит Zero-shot CLIP. Это говорит о том, что модель получает дополнительную "уверенность"при предсказании не только наиболее похожего класса, но и при получении нескольких значимых концептов на изображение.

6.1 Исследование абляции

Компоненты, введенные в наш фреймворк, приводят к снижению итоговой производительности по сравнению с линейным зондированием (см. table 2), обеспечивая при этом интерпретируемость, которую мы представляем в appendix D.2. Наш метод и его интуиция, заключающаяся в том, чтобы сделать внутреннее представление более разреженным, понятны, и он оказывается перспективным для извлечения значимых понятий. Тем не менее, "жесткая"(т.е. делающая выборки непосредственно одномоментными) выборка в контрастном Gumbel-Softmax loss скорее вредит производительности Sparse-CBM,

чем улучшает ее. В то же время Contrastive-CBM выполняет двойной softmax на CBL, который обеспечивает отсутствие разреженности и достигает худшего результата по точности, что опять же подтверждает нашу гипотезу о полезности разреженных внутренних представлений. И фреймворк CBM, и метод CMS полагаются на сгенерированный набор концептов. С концептами, которые лучше приспособлены к набору данных, мы показываем более высокие результаты по точности классификации, с другой стороны, мы хотим, чтобы концепты были более разнообразными, что эмпирически определено в section 4.3 и в предыдущей [25] работе, тогда нам не принципиально менять количество концептов вручную, мы предпочтем сохранить сгенерированные.

6.2 Ограничения

Помимо достигнутых показателей точности, мы сообщаем об основных ограничениях нашего фреймворка и алгоритма 1. LLM, предложенный во фреймворке CBM, по-прежнему не влияет на процесс классификации, то есть не поддерживает генерацию концептов во времени и работает только на **шаге 2**. Оба варианта CMS и CBM не модифицируют латентное пространство CLIP, что делает наш подход менее универсальным. Хотя мы вносим вклад в понимание разреженных CBL, эффективность моделей узких мест с также разреженными FC все еще не раскрыта. То же самое относится и к проблеме сквозного редактирования концепт-блотлека (см. section 1.2). Наконец, метод CMS может быть полезен в качестве дополнения к CLIP, но в долгосрочной перспективе не может конкурировать с подходами МД из-за ситуации, показанной в fig. 5.

Список литературы

[1] *Yu, Xiaowei.* NoisyNN: Exploring the Influence of Information Entropy Change in Learning Systems. — 2023.

[2] *Russakovsky, Olga.* ImageNet Large Scale Visual Recognition Challenge. — 2015.

[3] *Sun, Chen.* Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. — 2017.

[4] ImageNet: A large-scale hierarchical image database / Jia Deng, Wei Dong, Richard Socher et al. // 2009 IEEE Conference on Computer Vision and Pattern Recognition. — 2009. — Pp. 248–255.

[5] *Dosovitskiy, Alexey.* An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. — 2021.

[6] *He, Kaiming.* Deep Residual Learning for Image Recognition. — 2015.

[7] *Koh, Pang Wei.* Concept Bottleneck Models. — 2020.

[8] *Radford, Alec.* Learning Transferable Visual Models From Natural Language Supervision. — 2021.

[9] *Aljundi, Rahaf.* Contrastive Classification and Representation Learning with Probabilistic Interpretation. — 2022.

[10] *Reed, Scott.* A Generalist Agent. — 2022.

[11] *Chen, Lin.* ShareGPT4V: Improving Large Multi-Modal Models with Better Captions. — 2023.

[12] *Khosla, Prannay.* Supervised Contrastive Learning. — 2021.

[13] *Gao, Tianyu.* SimCSE: Simple Contrastive Learning of Sentence Embeddings. — 2022.

[14] *Srinivas, Aravind.* CURL: Contrastive Unsupervised Representations for Reinforcement Learning. — 2020.

[15] *Jia, Chao.* Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision. — 2021.

[16] *Li, Junnan.* BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. — 2022.

[17] *Zhai, Xiaohua.* LiT: Zero-Shot Transfer with Locked-image text Tuning. — 2022.

[18] *Chopra, S.* Learning a similarity metric discriminatively, with application to face verification / S. Chopra, R. Hadsell, Y. LeCun // 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). — Vol. 1. — 2005. — Pp. 539–546 vol. 1.

[19] *Zhai, Xiaohua.* Sigmoid Loss for Language Image Pre-Training. — 2023.

[20] *Wei, Jason.* EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. — 2019.

[21] *Kobayashi, Sosuke.* Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations. — 2018.

[22] *Fang, Hongchao.* CERT: Contrastive Self-supervised Learning for Language Understanding. — 2020.

[23] *Shen, Dinghan.* A Simple but Tough-to-Beat Data Augmentation Approach for Natural Language Understanding and Generation. — 2020.

[24] *Yuksekgonul, Mert.* Post-hoc Concept Bottleneck Models. — 2023.

[25] Label-free Concept Bottleneck Models / Tuomas Oikarinen, Subhro Das, Lam M. Nguyen, Tsui-Wei Weng // The Eleventh International Conference on Learning Representations. — 2023. https://openreview.net/forum?id=FlCg47MNvBA.

[26] *Kim, Been.* Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). — 2018.

[27] *Weng, Lilian.* Contrastive Representation Learning / Lilian Weng // *lilianweng.github.io.* — 2021. — May. `https://lilianweng.github.io/posts/2021-05-31-contrastive/`.

[28] The Caltech-UCSD Birds-200-2011 Dataset / Catherine Wah, Steve Branson, Peter Welinder et al. — 2011. — Jul.

[29] Places: A 10 Million Image Database for Scene Recognition / Bolei Zhou, Agata Lapedriza, Aditya Khosla et al. // *IEEE Transactions on Pattern Analysis and Machine Intelligence.* — 2018. — Vol. 40, no. 6. — Pp. 1452–1464.

[30] *Krizhevsky, Alex.* Learning Multiple Layers of Features from Tiny Images / Alex Krizhevsky. — 2009. — Pp. 32–33. `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.

[31] *Losch, Max.* Interpretability Beyond Classification Output: Semantic Bottleneck Networks. — 2019.

[32] Contextual Semantic Interpretability / Diego Marcos, Ruth Fong, Sylvain Lobry et al. // Proceedings of the Asian Conference on Computer Vision (ACCV). — 2020. — November.

[33] *LeCun, Yann.* Convolutional networks and applications in vision / Yann LeCun, Koray Kavukcuoglu, Clement Farabet // Proceedings of 2010 IEEE International Symposium on Circuits and Systems. — 2010. — Pp. 253–256.

[34] *Yang, Yue.* Language in a Bottle: Language Model Guided Concept Bottlenecks for Interpretable Image Classification. — 2023.

[35] *Schwalbe, Gesina.* Concept Embedding Analysis: A Review. — 2022.

[36] *Wong, Eric.* Leveraging Sparse Linear Layers for Debuggable Deep Networks / Eric Wong, Shibani Santurkar, Aleksander Madry //

Proceedings of the 38th International Conference on Machine Learning / Ed. by Marina Meila, Tong Zhang. — Vol. 139 of *Proceedings of Machine Learning Research*. — PMLR, 2021. — 18–24 Jul. — Pp. 11205–11216. https://proceedings.mlr.press/v139/wong21b.html.

[37] *Menon, Sachit*. Visual Classification via Description from Large Language Models / Sachit Menon, Carl Vondrick // *ICLR*. — 2023.

[38] *Oikarinen, Tuomas*. CLIP-Dissect: Automatic Description of Neuron Representations in Deep Vision Networks. — 2023.

[39] *Gao, Yuting*. PyramidCLIP: Hierarchical Feature Alignment for Vision-language Model Pretraining. — 2022.

[40] *Kazmierczak, Rémi*. CLIP-QDA: An Explainable Concept Bottleneck Model. — 2023.

[41] Cross-Modal Conceptualization in Bottleneck Models / Danis Alukaev, Semen Kiselev, Ilya Pershin et al. // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. — Association for Computational Linguistics, 2023. http://dx.doi.org/10.18653/v1/2023.emnlp-main.318.

[42] *Panousis, Konstantinos P.* DISCOVER: Making Vision Networks Interpretable via Competition and Dissection. — 2023.

[43] *Panousis, Konstantinos P.* Nonparametric Bayesian Deep Networks with Local Competition. — 2019.

[44] *Paszke, Adam*. PyTorch: An Imperative Style, High-Performance Deep Learning Library. — 2019.

[45] *Wolf, Thomas*. HuggingFace's Transformers: State-of-the-art Natural Language Processing. — 2020.

[46] *Reimers, Nils.* Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks / Nils Reimers, Iryna Gurevych // Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. — Association for Computational Linguistics, 2019. — 11. https://arxiv.org/abs/1908.10084.

[47] *Reimers, Nils.* Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation / Nils Reimers, Iryna Gurevych // Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing. — Association for Computational Linguistics, 2020. — 11. https://arxiv.org/abs/2004.09813.

[48] *Kingma, Diederik P.* Adam: A Method for Stochastic Optimization. — 2017.

[49] *Loshchilov, Ilya.* Decoupled Weight Decay Regularization. — 2019.

[50] *Brown, Tom B.* Language Models are Few-Shot Learners. — 2020.

[51] *Touvron, Hugo.* LLaMA: Open and Efficient Foundation Language Models. — 2023.

[52] *Speer, Robyn.* ConceptNet 5: A Large Semantic Network for Relational Knowledge / Robyn Speer, Catherine Havasi // *The people's web meets NLP, theory and applications of natural language processing.* — 2013. — 02. — Pp. 161–176.

[53] *Chen, Yihao.* DisCo-CLIP: A Distributed Contrastive Loss for Memory Efficient CLIP Training. — 2023.

[54] *Jang, Eric.* Categorical Reparameterization with Gumbel-Softmax. — 2017.

[55] *Maddison, Chris J.* The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables / Chris J. Maddison, Andriy Mnih, Yee Whye Teh // *CoRR.* — 2016. — Vol. abs/1611.00712. http://arxiv.org/abs/1611.00712.

[56] *Gumbel, Emil Julius.* Statistical Theory of Extreme Values and Some Practical Applications : A Series of Lectures / Emil Julius Gumbel. — 1954. https://api.semanticscholar.org/CorpusID:125881359.

[57] *Maddison, Chris J.* A* Sampling. — 2015.

[58] *Szegedy, Christian.* Rethinking the Inception Architecture for Computer Vision. — 2015.

[59] *Pereyra, Gabriel.* Regularizing Neural Networks by Penalizing Confident Output Distributions. — 2017.

[60] *Alexandridis, Konstantinos Panagiotis.* Long-tailed Instance Segmentation using Gumbel Optimized Loss. — 2022.

[61] *Chauhan, Kushal.* Interactive Concept Bottleneck Models. — 2023.

[62] *van der Maaten, Laurens.* Visualizing Data using t-SNE / Laurens van der Maaten, Geoffrey Hinton // *Journal of Machine Learning Research.* — 2008. — Vol. 9, no. 86. — Pp. 2579–2605. http://jmlr.org/papers/v9/vandermaaten08a.html.

A Аппендикс

Мы сравниваем наши архитектуры Sparse-, ℓ_1-, Contrastive-CBM с Post-hoc CBM [24] и LaBo [34] с точки зрения точности в задаче классификации изображений. В то время как интерпретируемость концептов сравнивается со свойствами магистральной мультимодальной модели в appendix D.2. Поскольку фреймворки Post-hoc также позволяют создать модель Concept Bottleneck над стандартной, мы оцениваем этот метод на модели CLIP-ViT-L/14 для справедливого сравнения, в то же время архитектуры LaBo построены с той же самой базовой моделью с помощью defalut. Кроме того, мы приводим результаты для линейного зондирования CLIP-ViT-L/14. В этом случае мы добавляем только один линейный слой после матрицы Image-Class, возвращаемой CLIP. Обновленные результаты можно посмотреть в table 2.

Настройка "full-supervised"в LaBo означает, что архитектура обучается на всех доступных изображениях. Мы отмечаем это в связи с тем, что рассматриваются режимы обучения [34] zero-shot и N-shot.

B Дополнительные эксперименты

В этом разделе мы сообщаем о дополнительных результатах оценки фреймворка CBM (раздел 4.3) и алгоритма CMS (раздел 4.2).

B.1 Обучение Concept Bottleneck Model

Мы приводим несколько графиков (figs. 6 to 8), наблюдаемых во время обучения Sparse-CBM на CIFAR10, CUB200 и ImageNet-1K.

Таблица 2: Конфигурации магистральных моделей. Пересечения указывают на размер модели с соответствующей конфигурацией.

B/32	L/14	Dataset
0.57GB	1.63GB	CIFAR10
0.58GB	1.64GB	CIFAR100
0.68GB	1.74GB	ImageNet
0.58GB	1.64GB	CUB200
0.61GB	1.67GB	Places365

C Дополнительные данные

D Визуализации

D.1 Анализ латентного пространства CLIP

В этом разделе мы представим дополнительные эксперименты с латентным пространством CLIP. Во-первых, используя CLIP, мы построили двумерную t-SNE карту вкраплений CIFAR10 вместе с их концептами и классами. Интересным результатом является то, что это пространство не просто разделено на два кластера: один соответствует текстовой модальности, а второй - визуальной, но и, если добавить проекцию случайных слов на t-SNE, мы наблюдаем ее пересечение с концептами. Соответствующий эксперимент представлен в fig. 11

С помощью метода кластеризации k-means мы строим различимые по CLIP кластеры (см. fig. 12).

Важным моментом в t-SNE-визуализации является то, что латентное пространство CLIP-подобных моделей строго делится на два кластера: один соответствует вкраплениям изображений, а второй - текстовой модальности. Интуиция, лежащая в основе МД, подсказывает, что распределение модальностей должно полностью отличаться от того, что наблюдается в fig. 11. Действительно, для удобства ин-

терпретации предполагается, что вкрапления изображений должны быть ближе к векторам соответствующих понятий и классов. Если это так, то простой алгоритм kNN сможет найти наиболее релевантные понятия, что в нашем случае неверно. Таким образом, мы выделяем нерешенную проблему построения моделей, которые будут изучать подобное латентное пространство сквозным образом.

D.2 Интерпретируемость концептов

Мы сравниваем возможности интерпретации frawework с базовыми свойствами извлечения признаков CLIP на подмножествах изображений из наборов данных Places365, CUB200, CIFAR10 и ImageNet. Мы выбираем как базовые изображения для наглядности, так и сложные, с большим количеством внутренних деталей. Для CLIP-подобных моделей мы регистрируем топ-к (k=10) наивысших оценок точечного продукта, а для наших фреймворковых архитектур - результаты работы Concept Bottleneck Layer. Отметим, что дальнейшие результаты в figs. 13 to 18 проведены с использованием магистральной модели CLIP-ViT-L/14. Мы определенно наблюдаем распыление активаций с помощью как Sparse-, так и ℓ_1-CBM. Также показано, что концептуальный слой Bottleneck Layer, обученный с контрастной целью, производит довольно схожие активации по сравнению с базовой моделью CLIP. В то же время итоговая точность становится выше при использовании разреженных и ℓ_1-моделей, что говорит о превосходстве методов, разрежающих внутренние слои модели.

В данной работе мы не приводим интерпретируемые матрицы поиска концептов, поскольку этот подход к классификации с узкими местами концептов не модифицирует базовую модель CLIP, поэтому она обладает теми же свойствами, что и figs. 13 to 18 (a).

D.3 Sparse-CBM ошибки

Вместе с итоговыми результатами классификации мы приводим матрицу путаницы лучшей Sparse-CBM, обученной на наборе данных

CUB200, которая достигает 80,02% точности в fig. 19. Следует заметить, что наиболее значительные ошибки модели закладываются в последней части меток. Сюда входят несколько схожих классов, таких как "черношапочный вирео "синеголовый вирео "филадельфийский вирео "красноглазый вирео "певчий вирео "белоглазый вирео"и "желтоголовый вирео". В целом, в таких разнообразных наборах данных, как CUB200, представлено множество схожих классов, поэтому имеет смысл рассмотреть возможность выделения понятий и подчеркнуть их различия между такими классами, как "красноглазый вирео"и "белоглазый вирео".

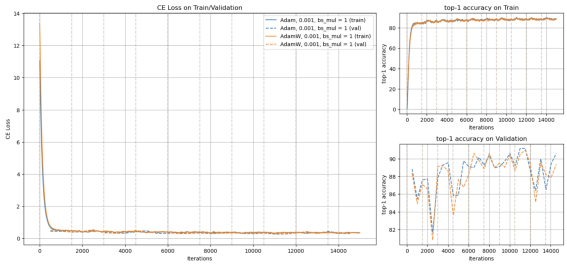


Рис. 6: Лучший результат Sparse-CBM на CIFAR10.

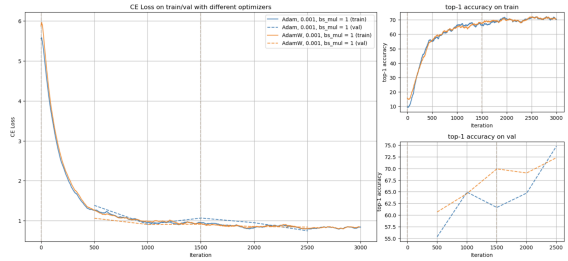


Рис. 7: Лучший результат Sparse-CBM на CUB200.

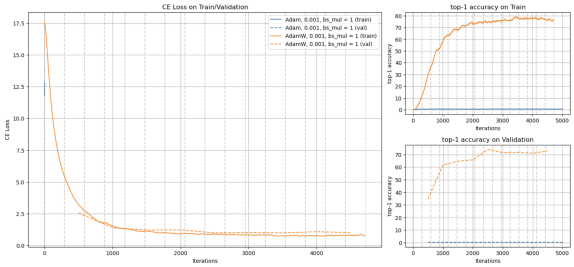


Рис. 8: Лучший результат Sparse-CBM на ImageNet-1K.

Рис. 9: Обзор режима обучения с использованием разреженного CBM на нескольких наборах данных. Мы наблюдаем аналогичные кривые потерь при обучении на всех данных. Но для ImageNet обучение начинается с меньшей скорости обучения.

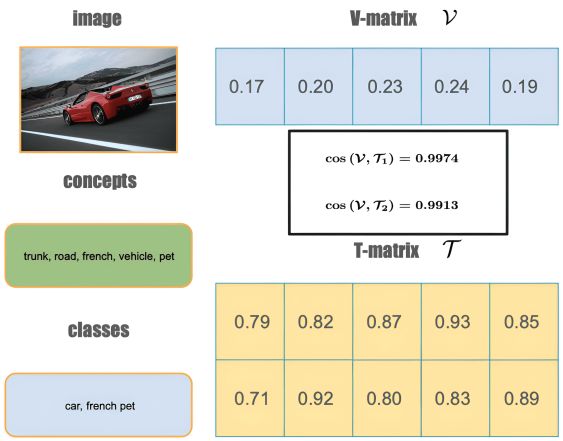


Рис. 10: Визуализация Concept Matrix Search *гипотезы* для простого случая 1 картинки, 5 концептов и 2 классов.

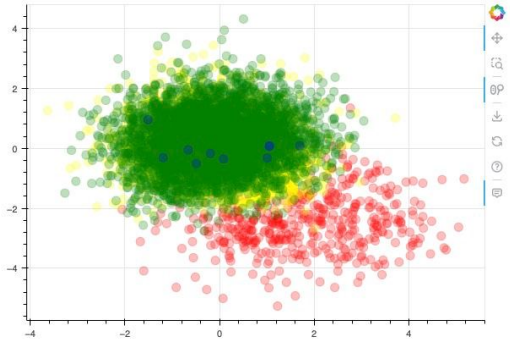


Рис. 11: Визуализация CIFAR10 t-SNE. Зеленые точки относятся к проекции понятий, синие - к классам, красные - к изображениям, а желтые - к случайным словам.

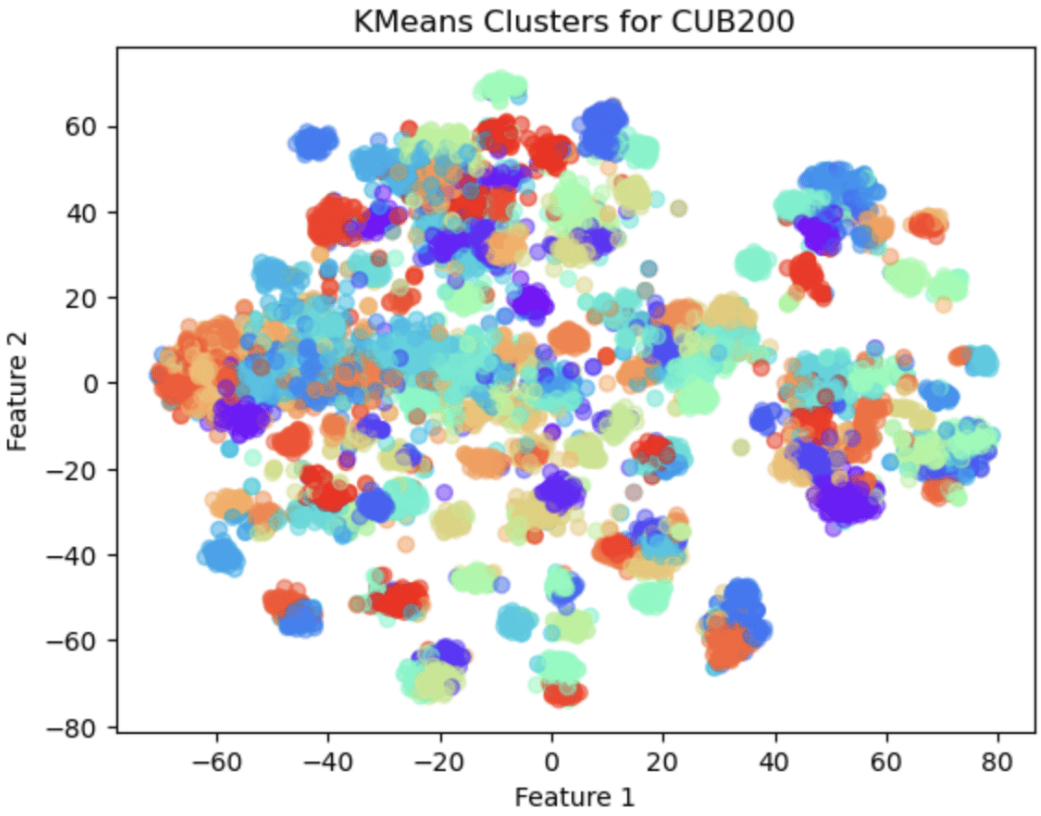
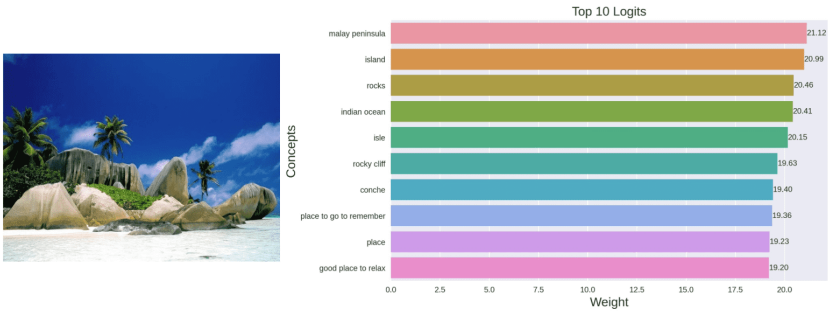
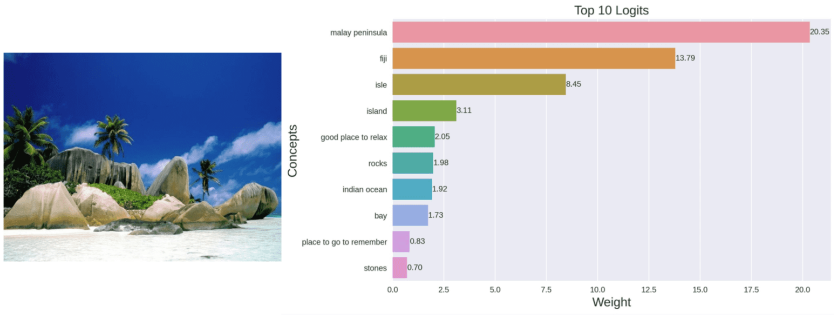


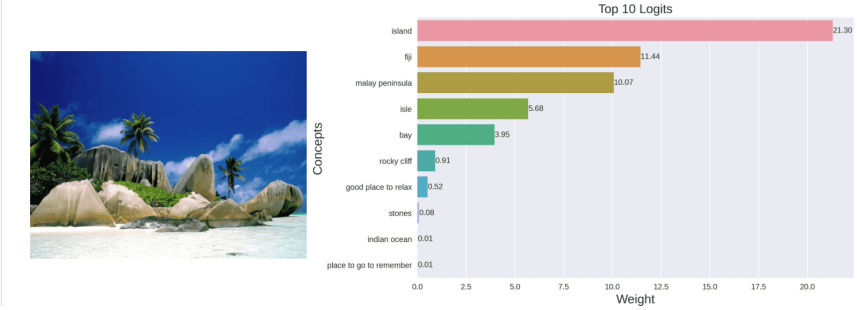
Рис. 12: Визуализауия k-means кластеров эмбеддингов картинок из CUB200 полученных с помощью CLIP.



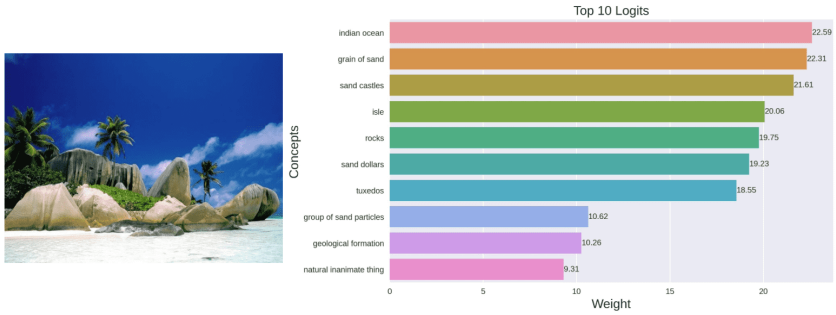
(a) Концепты, извлекаемые с помощью CLIP.



(b) Концепты, извлекаемые с помощью Sparse-CBM.

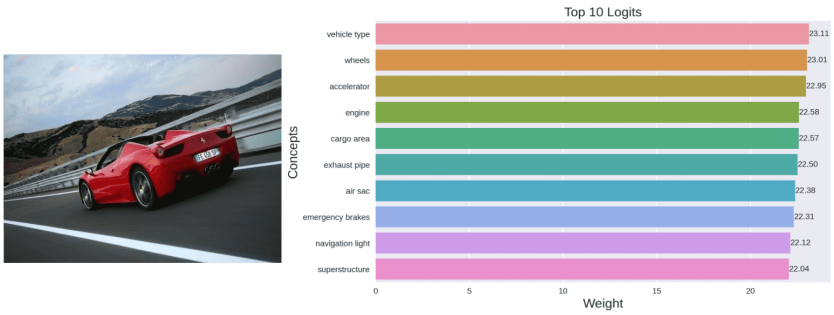


(c) Концепты, извлекаемые с помощью ℓ_1-CBM.

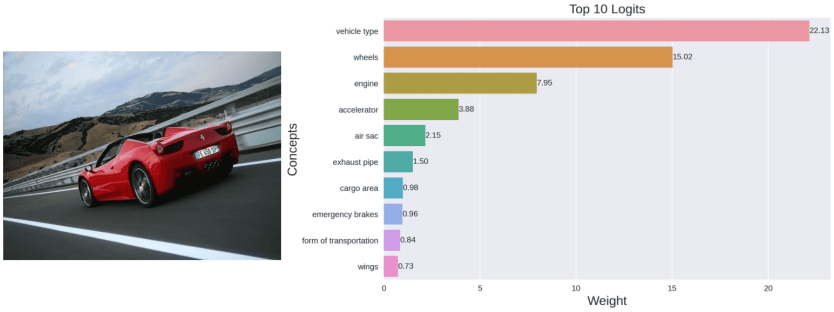


(d) Концепты, извлекаемые с помощью Contrastive-CBM.

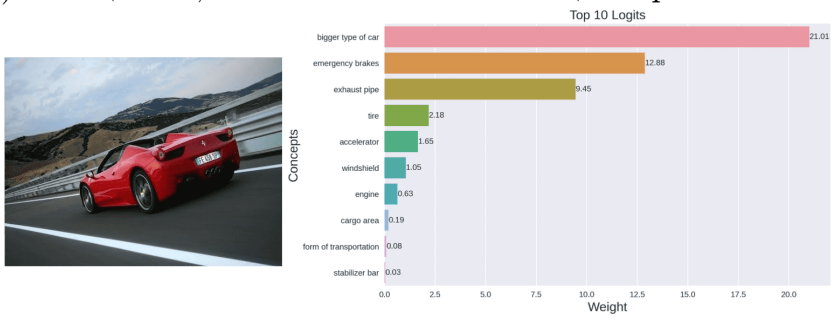
Рис. 13: Концепты, извлекаемые с помощью моделей, обученных на Places365.



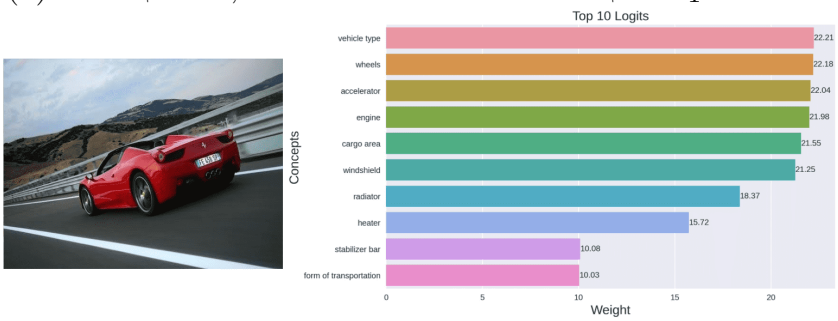
(a) Концепты, извлекаемые с помощью CLIP.



(b) Концепты, извлекаемые с помощью sparse-CBM.

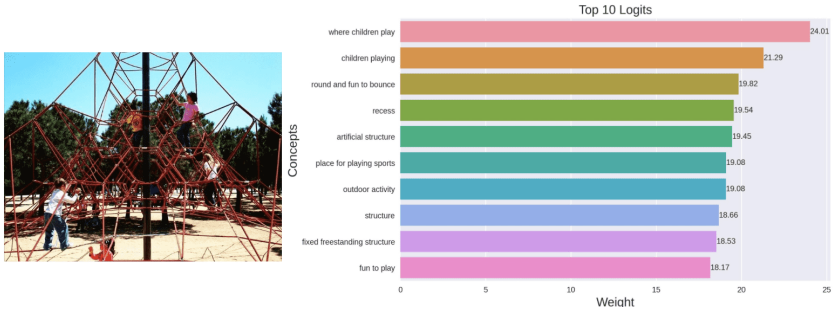


(c) Концепты, извлекаемые с помощью ℓ_1-CBM.

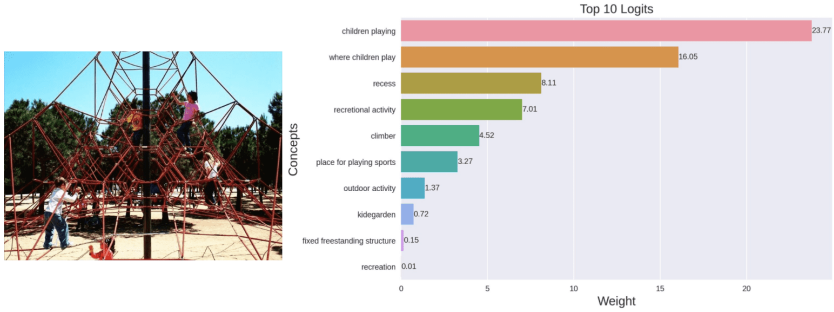


(d) Концепты, извлекаемые с помощью Contrastive-CBM.

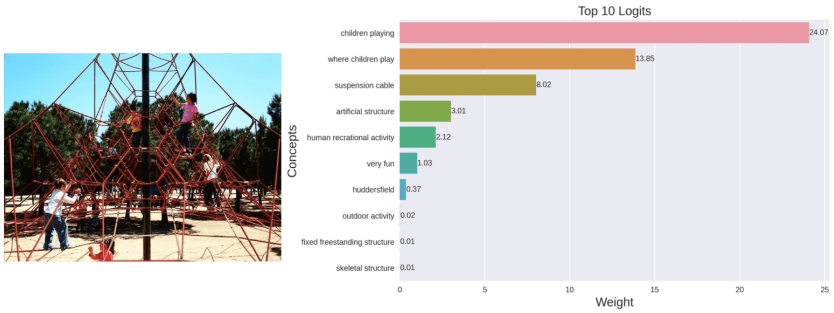
Рис. 14: Концепты, извлекаемые с помощью моделей, обученных на CIFAR10.



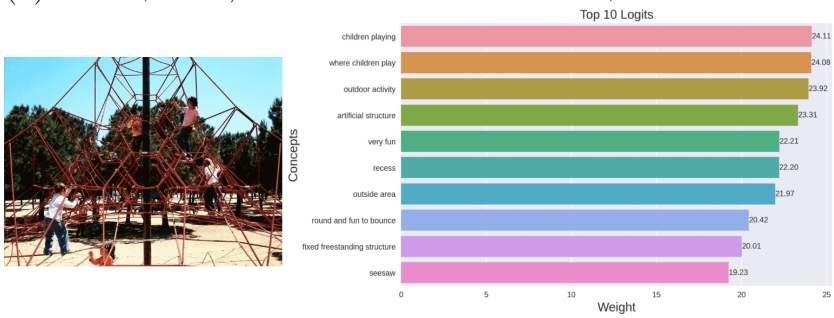
(a) Концепты, извлекаемые с помощью CLIP.



(b) Концепты, извлекаемые с помощью Sparse-CBM.

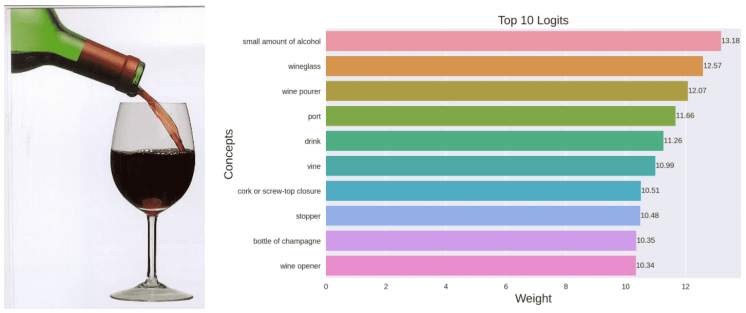


(c) Концепты, извлекаемые с помощью ℓ_1-CBM.

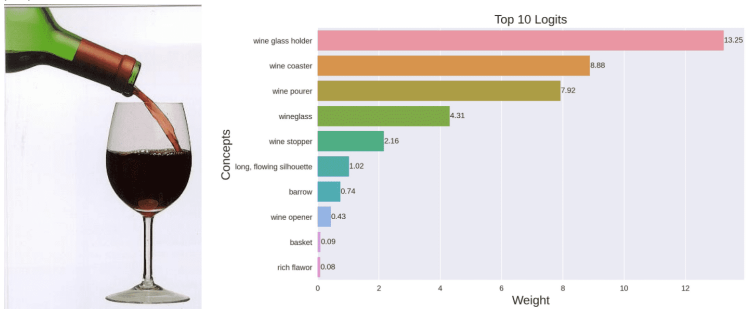


(d) Концепты, извлекаемые с помощью Contrastive-CBM.

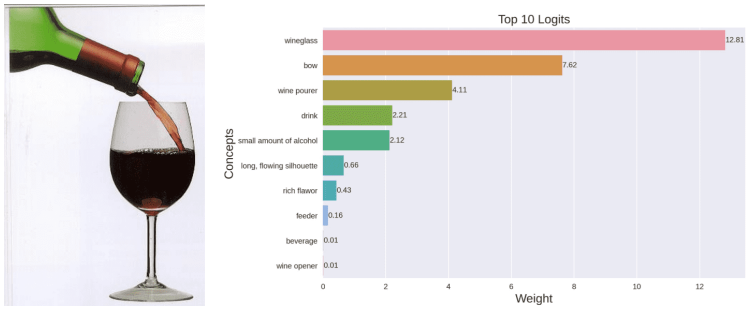
Рис. 15: Концепты, извлекаемые с помощью моделей, обученных на Places365.



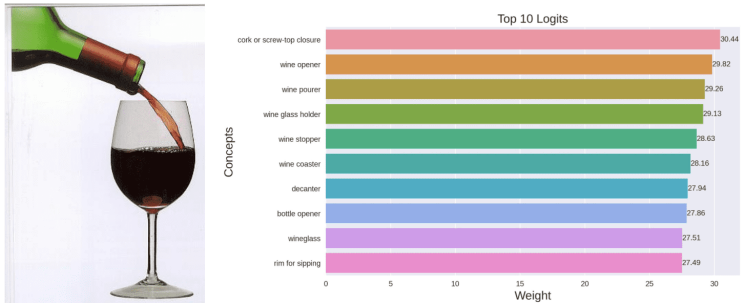
(a) Концепты, извлекаемые с помощью CLIP.



(b) Концепты, извлекаемые с помощью Sparse-CBM.

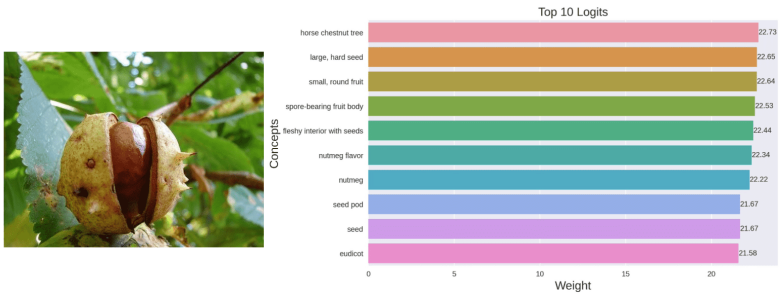


(c) Концепты, извлекаемые с помощью ℓ_1-CBM.

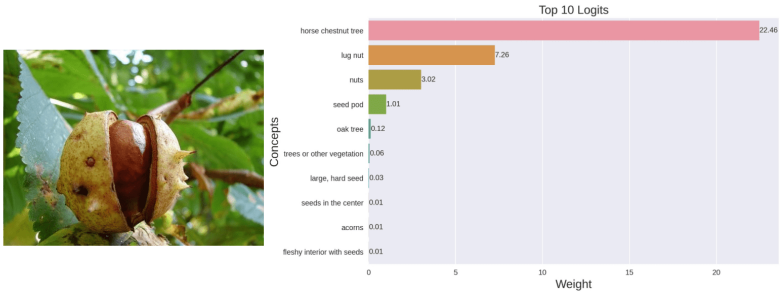


(d) Концепты, извлекаемые с помощью Contrastive-CBM.

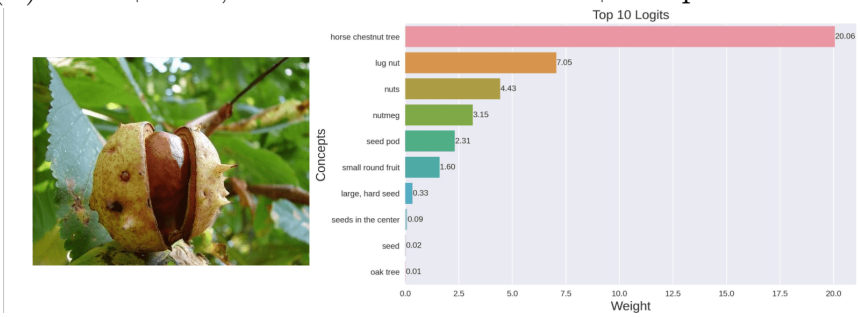
Рис. 16: Концепты, извлекаемые с помощью моделей, обученных на ImageNet.



(a) Концепты, извлекаемые с помощью CLIP.



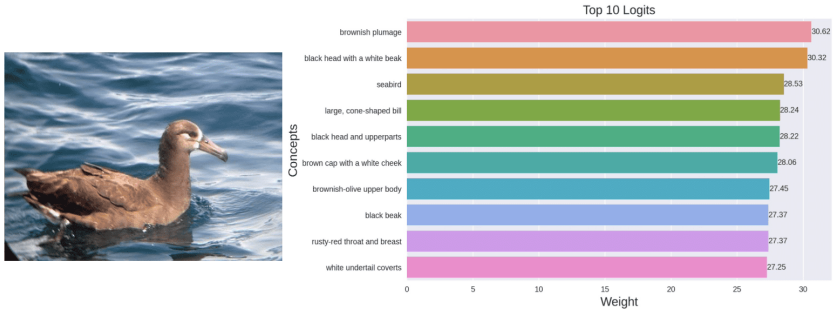
(b) Концепты, извлекаемые с помощью Sparse-CBM.



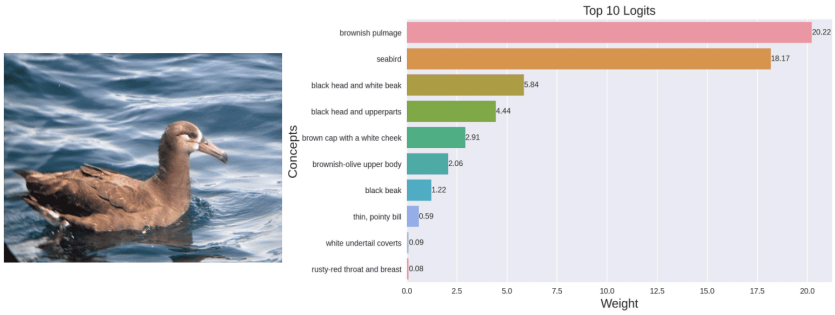
(c) Концепты, извлекаемые с помощью ℓ_1-CBM.

(d) Концепты, извлекаемые с помощью Contrastive-CBM.

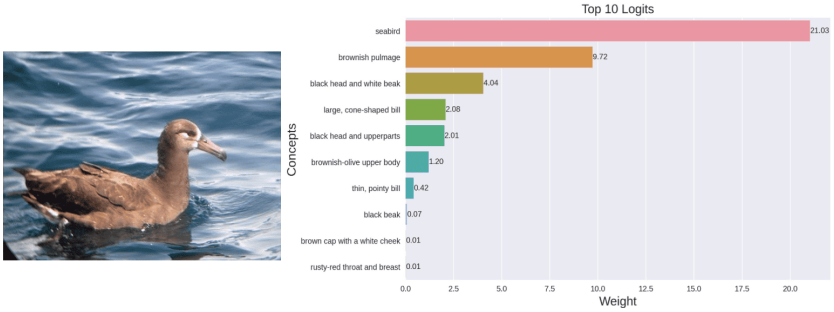
Рис. 17: Концепты, извлекаемые с помощью моделей, обученных на Places365.



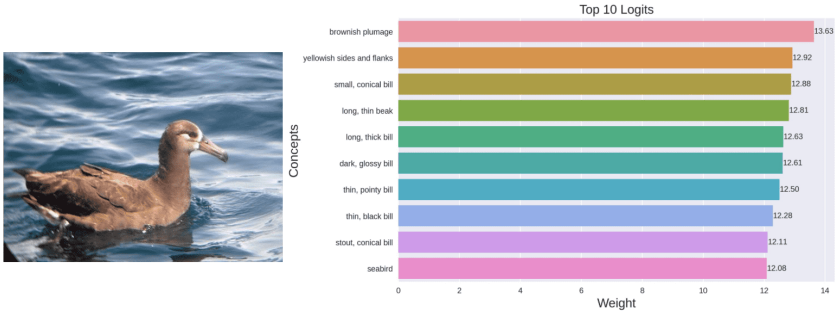
(a) Концепты, извлекаемые с помощью CLIP.



(b) Концепты, извлекаемые с помощью Sparse-CBM.



(c) Концепты, извлекаемые с помощью ℓ_1-CBM.



(d) Концепты, извлекаемые с помощью Contrastive-CBM.

Рис. 18: Концепты, извлекаемые с помощью моделей, обученных на CUB200.

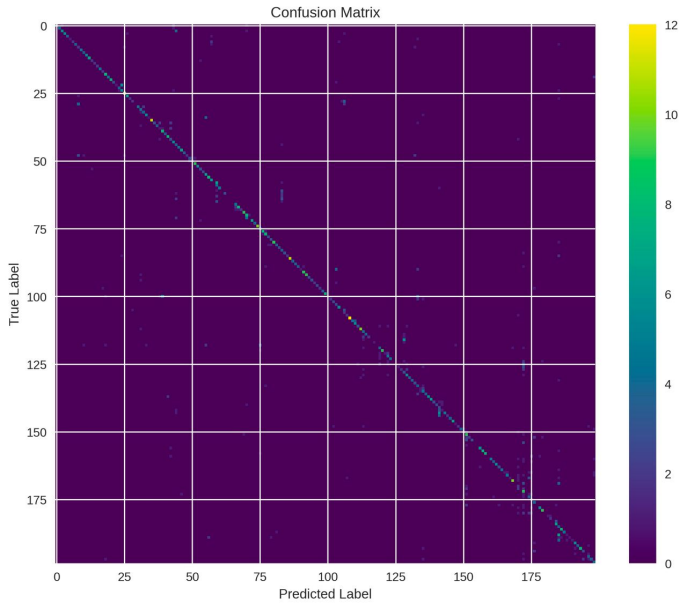


Рис. 19: Матрица ошибок Sparse-CBM на наборе данных CUB200.