ДОКЛАДЫ РОССИЙСКОЙ АКАДЕМИИ НАУК. МАТЕМАТИКА, ИНФОРМАТИКА, ПРОЦЕССЫ УПРАВЛЕНИЯ, 2022, том 508, с. 134–145

ПЕРЕДОВЫЕ ИССЛЕДОВАНИЯ В ОБЛАСТИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА И МАШИННОГО ОБУЧЕНИЯ

УДК 004.8

eco2AI: КОНТРОЛЬ УГЛЕРОДНОГО СЛЕДА МОДЕЛЕЙ МАШИННОГО ОБУЧЕНИЯ В КАЧЕСТВЕ ПЕРВОГО ШАГА К УСТОЙЧИВОМУ ИСКУССТВЕННОМУ ИНТЕЛЛЕКТУ

© 2022 г. С. А. Буденный[1,2,*], В. Д. Лазарев[2], Н. Н. Захаренко[1], А. Н. Коровин[2], О. А. Плосская[1], Д. В. Димитров[1], В. С. Ахрипкин[1], И. В. Павлов[1], И. В. Оселедец[2,3], И. С. Барсола[4], И. В. Егоров[4], А. А. Костерина[4], Л. Е. Жуков[5]

Представлено академиком РАН А.П. Кулешовым
Поступило 28.10.2022 г.
После доработки 28.10.2022 г.
Принято к публикации 01.11.2022 г.

На сегодняшний день в самых различных областях науки и производства возрастает значение искусственного интеллекта (ИИ), в частности, моделей глубокого обучения. Вместе с развитием вычислительных систем наблюдается экспоненциальный рост сложности моделей ИИ, увеличивается их энергопотребление в процессе обучения и инференса. В статье представляется библиотека на Python с открытым исходным кодом eco2AI, который поможет исследователям и аналитикам контролировать потребления энергии и эквивалентную эмиссию CO_2 моделей ИИ. В eco2AI делается акцент на точности отслеживания энергопотребления и правильном региональном учете эмиссии CO_2. Авторы библиотеки призывают исследовательское сообщество к поиску более энергоэффективных архитектур моделей ИИ, а также предлагают концепцию циклического снижения парниковых газов комбинацией концепций устойчивого развития и зеленого ИИ. Код библиотеки и документация размещены в репозитории Github под лицензией Apache 2.0 https://github.com/sb-ai-lab/Eco2AI.

Ключевые слова: ESG, ИИ, Устойчивое развитие, Углеводный след, Экология, Эмиссия CO_2, Парниковые газы

DOI: 10.31857/S2686954322070232

1. ВВЕДЕНИЕ

Несмотря на то, что глобальная повестка ESG (окружающая среда, социальное и корпоративное управление) руководствуется соглашениями, заключенными между странами [1], фактическое развитие принципов ESG происходит посредством внедрения корпоративных, исследовательских и академических стандартов. По этой причине многие компании начали разрабатывать свои стратегии ESG, создавать полноценные отделы, публиковать ежегодные отчеты по устойчивому развитию, выделять дополнительные средства на исследования, в т.ч. цифровых технологий и искусственного интеллекта.

При этом остается актуальной проблема прозрачной и объективной количественной оценки прогресса ESG в области охраны окружающей среды. Это имеет большое значение для ИТ-индустрии, поскольку уже около одного процента мировой электроэнергии потребляется облачными вычислениями и их доля продолжает расти [31]. Искусственный интеллект и машинное обучение (МО) являются важной частью современной ИТ-индустрии, это быстро развивающиеся технологии с огромным потенциалом для прорывного развития. Существует ряд способов, с помощью которых ИИ и МО могли бы смягчить экологические проблемы и антропогенное воздействие. В частности, их можно было бы использовать для генерации и обработки больших данных, для более точного изучения Земли и прогнозирования поведения окружающей среды в различных сценариях [43] с целью улучшения понимания экологических процессов и принятия более обоснованных решений. Существует также потенциал использования ИИ и МО для моделирования результатов вредоносных процессов для экологии, таких как вырубка лесов, эрозия почвы,

[1] *Sber AI Lab, Москва, Россия*
[2] *Институт искусственного интеллекта AIRI, Москва, Россия*
[3] *Сколковский институт науки и технологий, Москва, Россия*
[4] *Sber ESG, Москва, Россия*
[5] *Национальный исследовательский университет "Высшая школа экономики", Москва, Россия*
*****E-mail: sanbudenny@sberbank.ru*

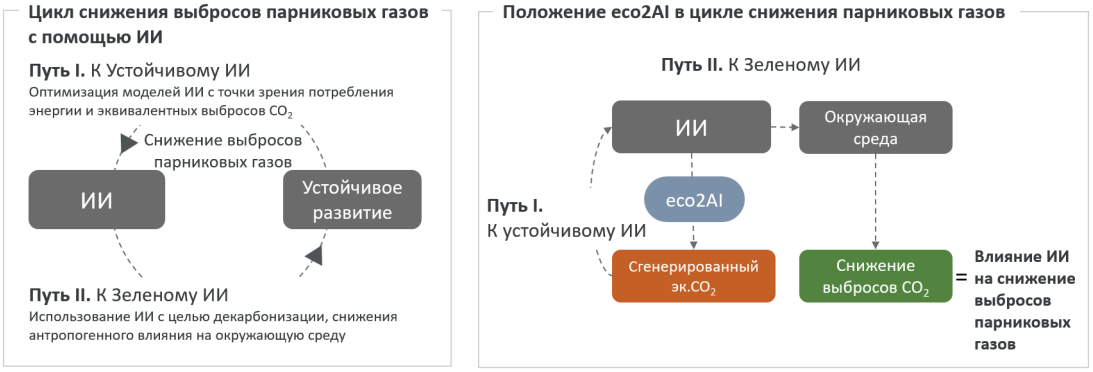

Рис. 1. Обобщенная схема снижения выбросов парниковых газов с помощью ИИ (схема слева), роль *eco2AI* в этой концепции (схема справа).

наводнения, увеличение содержания парниковых газов в атмосфере и т.д. В конечном счете, эти технологии обладают огромным потенциалом для улучшения нашего понимания окружающей среды и контроля над ней.

В настоящее время разрабатывается ряд решений на основе искусственного интеллекта для достижения углеродной нейтральности в рамках концепции "Зеленого искусственного интеллекта". Конечной целью этих решений является сокращение эмиссии парниковых газов (ПГ). В действительности, искусственный интеллект может помочь уменьшить последствия климатического кризиса, например, путем проектирования интеллектуальных систем, развития инфраструктуры с низким уровнем эмиссии и моделирования изменений климата [8]. Также крайне важно учитывать эмиссию CO_2, генерируемую самим ИИ в результате обучения моделей и их применения. Искусственный интеллект развивается в сторону моделей с большей вычислительной сложностью и потреблением электроэнергии и, как следствие, возрастающим в связи с этим косвенным эквивалентным углеродным следом. Экологическое воздействие искусственного интеллекта является основным фактором, который необходимо учитывать при определении возможных рисков. Чтобы модели ИИ/МО были экологически устойчивыми, они должны быть оптимизированы не только с точки зрения точности прогнозирования, но и с точки зрения потребления энергии и воздействия на окружающую среду. Таким образом, отслеживание воздействия ИИ на окружающую среду является первым шагом на пути к концепции устойчивого ИИ. Четкое понимание воздействия ИИ на окружающую среду мотивирует сообщество ИТ специалистов к поиску оптимальных архитектур, потребляющих меньше вычислительных ресурсов [38].

В статье представлена концепция цикла снижения выбросов парниковых газов с помощью ИИ, которая описывает возможности ИИ для достижения целей устойчивого развития (рис. 1). Концепция устойчивого развития обусловливает спрос на более энергоэффективные модели ИИ (рис. 1, путь "К устойчивому ИИ"). С другой стороны, ИИ создает дополнительные возможности для достижения целей устойчивого развития, и мы предлагаем назвать этот путь "К зеленому ИИ". Роль библиотеки *eco2AI* в этом цикле указана в правой части рис. 1. Во-первых, *eco2AI* мотивирует оптимизировать саму технологию ИИ. Во-вторых, если ИИ направлен на снижение выброса ПГ, то общий эффект следует оценивать с учетом генерируемого экв. CO_2, по крайней мере во время обучения модели (и в лучшем случае во время инференса модели). В рамках этой статьи авторы ограничись рассмотрением только пути "К устойчивому ИИ" (см. примеры в главе "Эксперименты").

Научный вклад работы:

• Во-первых, представлена *eco2AI*, библиотека Python с открытым исходным кодом, разработанная для оценки эквивалентной эмиссии CO_2 во время обучения моделей ИИ.

• Во-вторых, описана роль *eco2AI* в контексте концепции цикла снижения выбросов парниковых газов с помощью искусственного интеллекта.

• В-третьих, продемонстрированы примеры использования *eco2AI* в качестве средства оптимизации сложных fusion ИИ моделей.

Статья состоит из следующих разделов: в разделе **2** рассматриваются существующие решения для контроля уровня эквивалентного CO_2 при обучении моделей ИИ и описываются отличия от библиотеки *eco2AI*. В разделе **3** представлена методология вычисления эквивалентного CO_2. В разделе **4** продемострированы варианты использования

библиотеки. Наконец, в разделе **5** подводятся итоги работы. В приложении приводятся примеры использования библиотеки в коде.

2. СМЕЖНЫЕ ИССЛЕДОВАНИЯ

В этой главе описываются современные методы оценки косвенной эквивалентной эмиссии CO_2 для моделей ИИ, приводится краткое описание существующих пакетов с открытым исходным кодом.

2.1. Практика контроля эквивалентной эмиссии CO_2 связанной с работой моделей ИИ

С момента появления моделей глубокого обучения в 2012 г. их сложность росла в геометрической прогрессии, количество параметров удваивалось каждые 3–4 мес и достигло более триллиона параметров в 2022 г. Наиболее известными моделями являются BERT-Large (октябрь 2018 г., 3.4×10^8), GPT-2 (2019 г., 1.5×10^9), T5 (октябрь 2019 г., 1.1×10^{10}), GPT-3 (2020 г., 1.75×10^{11}), Megatron Turing (2022 г., 5.30×10^{11}), Switch Transformer (2022 г., 1.6×10^{12}).

На накопление, разметку, хранение, обработку и использование данных в течение срока их жизни от генерации до утилизации затрачивается значительное число ресурсов, масштаб которых можно оценить на примере инфраструктуры компании Amazon [24]. При этом их эффективный мониторинг важен для разработки сводов правил и законодательства [20].

В [38] проведено крупномасштабное исследование, направленное на количественную оценку приблизительных экологических издержек, связанных с обучением моделей ИИ, широко используемых для задач обработки текстов на естественном языке (NLP). Среди рассмотренных архитектур глубокого обучения таких, как Transformer, ELMo, BERT, NAS, GPT-2 оценивалось комбинированное энергопотребление GPU, CPU и DRAM, скорректированное на показатель эффективности использования энергии (PUE), специфического для конкретного центра обработки данных. Энергопотребление CPU и GPU определялось специализированными программными пакетами: Intel Running Average Power Limit и NVIDIA System Management. Произведением общего потребления энергии и коэффициента эмиссии углерода пересчитывают энергию в косвенную эмиссию CO_2. Авторы подсчитали, что углеродный след для обучения базового BERT составляет около 652 кг, что сравнимо с эмиссией CO_2 при авиаперелете “Нью-Йорк <–> Сан-Франциско” на одного пассажира.

Изучена возможность повышения энергоэффективности моделей NLP (T5, Meena, GShard, Switch Transformer, GPT-3) [30]. Показана возможность повышения энергоэффективности при обучении моделей нейронных сетей с помощью ряда методов, таких как: sparsely activating DL; distillation techniques [22]; pruning, quantization, efficient coding [19]; fine-tuning и transfer-learning [9]; обучение крупных моделей в конкретном регионе с низким энергопотреблением, использование облачных центров обработки данных, оптимизированных с точки зрения энергопотребления. Авторы ожидают, что принятие во внимание данных мер может сократить углеродный след в 10^2–10^3 раз.

2.2. Обзор существующих библиотек для отслеживания углеродного следа моделей ИИ

К настоящему времени разработан ряд библиотек для отслеживания косвенной эквивалентной эмиссии CO_2, связанной с обучением моделей ИИ (см. табл. 1).

Cloud Carbon Footprint [5] – это приложение, которое оценивает потребление энергии и углеродный след поставщиков общедоступных облачных сервисов. Приложение также предоставляет оценки как потребления энергии, так и углеродного следа для всех типов облачных сервисов, с возможностью детализации эмиссии по поставщику услуг, учетной записи, запущенной службе и периоду времени. Кроме того, для сервисов AWS и Google Cloud предоставляются рекомендации по экономии денег и минимизации эмиссии CO_2, прогнозируется экономия средств, а фактические сэкономленные ресурсы отображаются в посаженных деревьях. При этом для датацентров потребление энергии измеряется не на уровне среднего значения, а с использованием точных данных о реальной нагрузке на сервер в ходе его работы. Библиотека позволяет регистрируемыми показателями расширять существующие базы данных систем выставления счетов, конвейеров данных и систем мониторинга.

CodeCarbon [6] – это пакет Python для отслеживания эмиссии углерода, производимого при выполнении любого кода Python – от простых алгоритмов до глубоких нейронных сетей. CodeCarbon учитывает вычислительную инфраструктуру, местоположение, нагрузку на систему и время исполнения кода. Библиотека также позволяет проводить сравнение с эмиссией от обычных видов транспорта.

Carbontracker [4] – это пакет для отслеживания и прогнозирования энергопотребления и углеродного следа при обучении моделей глубокого обучения. Пакет направлен на использование прогноза энергопотребления для проактивного сокращения эмиссии CO_2. Например, обучение модели может быть остановлено по решению пользователя при превышении прогнозируемого

Таблица 1. Функции библиотек с открытым исходным кодом для оценки эквивалентной эмиссии CO_2 при обучении моделей ИИ

Библиотека	Cloud Carbon Footprint	Code Carbon	Carbon Tracker	Experimental Impact Tracker	Tracarbon	Green Algorithms	eco2AI
Общие сведения							
Первый выпуск	2020	2020	2020	2019	2022	2021	2022
Лизензия	Apache 2.0	MIT	MIT	MIT	Apache 2.0	CC-BY-4.0	Apache 2.0
Региональный коэффициент эмиссии	✓	✓	–	✓	✓	✓	✓*
Совместимая ОС							
Linux	✓	✓	✓	✓		✓	✓
Windows	✓	✓	✓			✓	✓
MacOS	✓	✓	✓	✓	✓	✓	✓
Совместимое оборудование							
RAM	✓	✓	✓	✓	✓	✓	✓
CPU	✓	✓	Неизвестно	✓	✓	✓	✓**
GPU	✓	✓	✓	✓	✓	✓	✓
Дополнительно							
Кодирование данных***							✓
WEB интерфейс	✓	✓				✓	

*База данных содержит коэффициенты интенсивности эмиссии для 365 территорий, в т.ч. данные по регионам Австралии ([13], [36]), Канады ([13], [40]), России ([34], [12]) и США ([13], [41]).

** *eco2AI* содержит базу данных CPU, состоящую из 3279 моделей от компаний Intel и AMD.

*** важно в случаях, когда необходимо подтвердить подлинность данных.

экологического ущерба. Библиотека поддерживает множество различных сред и платформ, таких как кластеры, настольные компьютеры и ноутбуки Google Colab, что позволяет работать по принципу plug-and-play [2].

Experiment impact tracker [16] – программный пакет, предоставляющий информацию об энергетическом, вычислительном и углеродном следе моделей машинного обучения. Он обладает следующими функциями: извлечение информации об устройствах CPU и GPU, определение времени начала и окончания эксперимента, учет региона оборудования, на котором проводится эксперимент (по IP-адресу), средняя интенсивность эмиссии углерода в регионе, расчет памяти и частоты процессора в реальном времени [20].

Green Algorithms [17] – это онлайн-инструмент, который позволяет пользователю оценивать и сообщать об углеродном следе в результате вычислений. Он интегрируется с вычислительными процессами и не взаимодействует с существующим кодом, а также учитывает модель CPU, GPU, облачных вычислений, локальных серверов и настольных компьютеров [26].

Tracarbon [39] – это Python библиотека, которая отслеживает энергопотребление устройства и рассчитывает углеродный след. Она автоматически определяет местоположение, модель CPU и GPU и может использоваться в качестве интерфейса командной строки (CLI) с предопределенными или рассчитанными с помощью API (интерфейс прикладного программирования) пользовательскими метриками.

При схожести *eco2AI* с описываемыми библиотеками в ней сделан акцент на следующем: учитываются только системные процессы, связанные непосредственно с обучением моделей (во избежание завышения оценки); база данных региональных коэффициентов интенсивности эмиссии (включено 365 территориальных объек-

тов) и база данных энергопотребления CPU (3279 моделей).

3. МЕТОДОЛОГИЯ

В главе рассматриваются вопросы расчета потребления электроэнергии, коэффициента эмиссии, эквивалентной эмиссии CO_2.

3.1. Расчет энергопотребления

Для расчета энергопотребления вычислительной системы необходимо оценить энергетический вклад каждого аппаратного блока [20]. В библиотеке *eco2AI* оценивается энергия, потребляемая графическим процессором (GPU), центральным процессором (CPU) и памятью (RAM) в силу их наибольшего вклада в энергопотребление среди всех аппаратных блоков. В процессе измерения пренебрегается вкладом краевых эффектов, связанных с завершающимися процессами, из-за их относительно небольшого влияния на общее энергопотребление. Также не учитывается энергопотребление систем хранения (SSD, HDD), так как они не имеют прямой связи с активностью процесса модели, но скорее всего о процессе постоянного хранения данных. Энергопотребление системы измеряется в джоулях (Дж), но чаще киловатт-часах (кВт·ч) – единице энергии, равной одному киловатту мощности, поддерживаемой в течение одного часа.

GPU. Библиотека *eco2AI* работает с GPU производства NVIDIA. Функционал обеспечивается библиотекой *Pynvml*, в которой реализован интерфейс Python для функций управления и мониторинга графических процессоров. Эта оболочка Python для библиотеки *nvml* от NVIDIA позволяет обнаруживать большинство GPU устройств NVIDIA, а также отслеживать количество активных устройств, их имена, используемую память, температуру, максимальную мощность (энергопотребление GPU может немного превышать это значение) и текущее энергопотребление каждого устройства. Для корректного обнаружения GPU требуется установка CUDA. Общее энергопотребление всех активных GPU E_{GPU} (кВт · ч) равно произведению энергопотребления графических процессоров на время их работы:

$$E_{GPU} = \int_0^T P_{GPU}(t)dt,$$

где P_{GPU} – суммарное энергопотребление(кВт) всех графических процессоров, определяемое функционалом *Pynvml*, T – время работы графических процессоров (ч). Если трекер не обнаруживает ни одного графического процессора, то энергопотребление GPU считается равным нулю.

CPU. Для мониторинга энергопотребления CPU использовался Python модуль *psutil*. Важным акцентом является то, что во избежание переоценки, в *eco2AI* реализован функционал, фильтрующий все фоновые процессы, библиотека учитывает только текущий процесс, связанный с обучением модели. Процент загрузки CPU определяется соотношением процента использования CPU и количества ядер. На данный момент создана самая полная база данных, содержащая 3279 уникальных процессоров для моделей Intel и AMD. Каждому наименованию модели CPU соответствует значение расчетной тепловой мощности (TDP), которое эквивалентно потребляемой мощности при длительных нагрузках. Суммарное энергопотребление всех активных процессорных устройств E_{CPU} (кВт · ч) равно произведению потребляемой мощности CPU устройств на время их загрузки $E_{CPU} = TDP\int_0^T W_{CPU}(t)dt$, где TDP – эквивалентная удельная мощность модели CPU при длительной нагрузке (кВт), W_{CPU} – суммарная загрузка всех процессоров. Если трекер не может определить ни одно процессорное устройство, энергопотребление процессора устанавливается равным 100 Вт [27].

RAM. Оперативная память является важным источником потребления энергии в современных вычислительных системах, особенно когда необходимо выделить или обработать значительный объем данных. Однако учет энергопотребления оперативной памяти проблематичен, так как ее энергопотребление сильно зависит от режима работы с данными: чтение, запись или хранение. В *eco2AI* энергопотребление RAM считается пропорциональным используемому количеству памяти и рассчитывается следующим образом: $E_{RAM} = 0.375\int_0^T M_{RAM_i}(t)dt$, где E_{RAM} – потребляемая энергия оперативной памяти (кВт · ч), M_{RAM_i} – используемая память (Гб), измеренная с помощью *psutil*, а 0.375 Вт/Гб – расчетное удельное энергопотребление модулей DDR3, DDR4 [27].

3.2. Региональный коэффициент интенсивности эмиссии CO_2

Эмиссия CO_2, связанная с производством электроэнергии, в значительной степени различается между странами и регионами. Для учета региональной зависимости эффективности эмиссии используют коэффициент интенсивности эмиссии γ, который определяется как масса выбрасываемого CO_2, выраженной в кг, на каждый мегаватт-час (МВт · ч) производимой электроэнергии. Коэффициент интенсивности эмиссии опреде-

Таблица 2. Коэффициенты интенсивности эмиссии CO_2 для некоторых регионов

Страна	ISO-Alpha-2 code	ISO-Alpha-3 code	UN M49 code	Коэффициент интенсивности эмиссии CO_2, кг/МВт · ч
Канада	CA	CAN	124	120.49
Франция	FR	FRA	250	67.53
Индия	IN	IND	356	625.57
Парагвай	PY	PRY	600	23.92
Замбия	ZM	ZMB	894	120.78

ляется региональным энергетическим балансом: $\gamma = \sum_i f_i e_i$, где i – индекс, относящийся к i-му источнику энергии (например, уголь, возобновляемые источники энергии, нефть, газ и т.д.), f_i – доля i-го источника энергии для конкретного региона, e_i – эмиссия, производимая сожжением килограмма массы этого источника энергии. Энергетический баланс в свою очередь определяется структурой производства электроэнергии, географическим положением, используемым топливом и технологическими процессами. Следовательно, чем выше доля возобновляемой энергии, тем меньше суммарный коэффициент интенсивности эмиссии. В противном случае большая доля углеводородных энергоресурсов в балансе приводит к более высокому значению коэффициента интенсивности эмиссии.

Библиотека *eco2AI* включает в себя базу данных коэффициентов интенсивности эмиссии для 365 регионов на основе общедоступных данных по 209 странам [11], а также региональных данных по таким странам, как Австралия ([13], [36]), Канада ([13], [40]), России ([34], [12], [28]) и США ([13], [41]). В настоящее время это самая большая база данных среди рассмотренных трекеров, что делает оценку энергопотребления более точной.

База данных имеет следующую структуру: название страны, код ISO-Alpha-2, код ISO-Alpha-3, код UN M49 и значение коэффициента интенсивности эмиссии. Коэффициенты интенсивности эмиссии CO_2 для некоторых регионов с различным энергетическим балансом приведен в табл. 2. Библиотека *eco2AI* автоматически определяет страну и регион пользователя по IP и находит в базе данных соответствующий им коэффициент эмиссии CO_2. Если коэффициент по какой-либо причине автоматически не определен, он устанавливается равным 436.5 кг/МВт · ч, среднее значение по миру [11]. В *eco2AI* можно вручную задать регион, страну и значение коэффициента интенсивности эмиссии.

3.3. Эквивалентное значение эмиссии CO_2

Наконец, эквивалентное значение эмиссии CO_2(кг), образующегося при обучении моделей, определяется путем умножения общего энергопотребления CPU, GPU и RAM на коэффициент эмиссии γ (кг/кВт · ч) и коэффициент PUE:

$$CF = \gamma \cdot PUE \cdot (E_{CPU} + E_{GPU} + E_{RAM}),$$

PUE – эффективность энергопотребления датацентра, необходимая, если процесс обучения выполняется в облаке. В *eco2AI* PUE является – опциональным задаваемым вручную параметром по умолчанию, равным 1.

4. ЭКСПЕРИМЕНТАЛЬНАЯ ЧАСТЬ

В этом разделе представлены эксперименты по отслеживанию эквивалентной эмиссии CO_2 с помощью *eco2AI* при обучении Malevich и Kandinsky. Malevich и Kandinsky – это большие мультимодальные text2image модели [18] с 1.3 миллиардом и 12 миллиардами параметров соответственно, способные генерировать произвольные изображения по текстовой строке.

В работе представлены результаты дообучения моделей Malevich и Kandinsky на наборе данных Emojis [37] и обучения Malevich с использованием оптимизированной функцией активации GELU [21].

4.1. Дообучение мультимодальных моделей

В этом разделе представлены примеры использования *eco2AI* для мониторинга дообучения моделей Malevich и Kandinsky (например, CO_2, кг; мощность, кВт·ч) на датасете Emojis. Malevich и Kandinsky – это мультимодальные трансформеры, которые обучаются условному распределе-

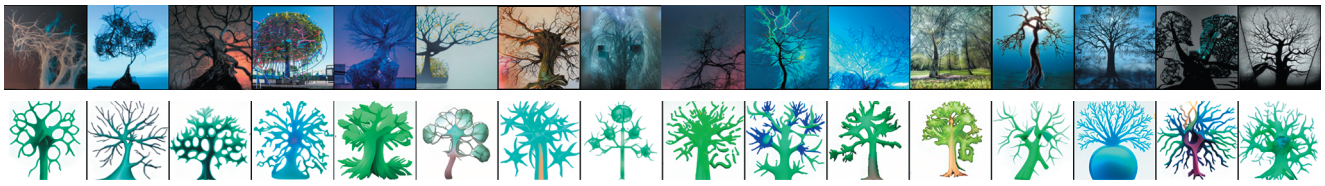

Рис. 2. Генерация изображений Malevich (сверху) vs Emojich XL (снизу) при вводе текста “Дерево в виде нейрона”.

Рис. 3. Генерация изображений Kandinsky (сверху) vs Emojich XXL (снизу) при вводе текста “Зеленый искусственный интеллект”.

нию изображений. Точнее, они авторегрессивно обрабатывают токены текста и изображения как единый поток данных (см., например, DALL-E [33]. Эти модели представляют собой декодеры-трансформеры [42] с 24 и 64 слоями, 16 и 60 attention heads, с размерностями скрытого пространства 2048 и 3840 соответственно и функцией активации GELU.

И Malevich, и Kandinsky работают со 128 текстовыми токенами, которые генерируются из текстовых входных данных с помощью токенизатора YTTM [44], и 1024 графическими токенами, которые получаются при кодировании входного изображения с помощью генеративно-состязательной сети Sber-VQGAN (предварительно обученный VQGAN [15] с Gumbel Softmax Relaxation [25].

Набор данных emoji [35] для дообучения содержит 2749 уникальных иконок emoji и 1611 уникальных текстов, которые были собраны с помощью парсинга веб-сайтов (разница в количествах связана с тем, что есть наборы, внутри которых смайлики отличаются только цветом, причем некоторые элементы являются омонимами).

Malevich и Kandinsky были обучены с точностями fp16 и fp32 соответственно. В обоих экспериментах используется оптимизатор Адам (8-бит) [7]. Эта реализация уменьшает объем памяти графического процессора, необходимой для хранения градиента. Выбраны следующие параметры обучения: начальное значение коэффициента скорости обучения(lr) 4×10^{-7}, максимальное значение – 10^{-5} и конечное значение – 2×10^{-8}. Модели обучались на 40 эпохах с коэффициентом изменения lr 0.3, размером батча 4 для Malevich и размером батча 12 для Kandinsky, с большим коэффициентом потери (loss coefficient) для изображения, равным 1000, и замороженными слоями с feed forward и attention.

Модели Malevich и Kandinsky обучались на 1 GPU Tesla A100 (80 ГБ) и 8 GPU Tesla A100 (80 ГБ) соответственно. Стоит отметить, что для обучения модели Kandinsky использовался оптимизатор распределенной модели DeepSpeed [45]. Исходный код, используемый для дообучения Malevich, доступен на Kaggle [14].

Результаты дообучения Malevich и Kandinsky называются Emojich XL и Emojich XXL соответственно. Сравнение результатов генерации Malevich и Emojich XL и Kandinsky и Emojicha XXL на некоторых текстовых токенах (см. рис. 2 и 3) позволяет визуально оценить качество дообучения (стиль сгенерированных изображений подстраивается под стиль emoji). Генерация изображения начинается с текстовой строки, описывающей ожидаемое содержимое. Когда токенизированный текст передается в Emojich, модель автоматически генерирует оставшиеся токены изображения.

Каждый токен изображения выбирается поэлементно из предсказанного полиномиального распределения вероятностей латентого пространства изображения с использованием nucleus sampling top-p и top-k с температурой [23] в качестве алгоритма декодирования. Изображение получается из сгенерированной последовательности скрытых векторов декодером Sber-VQGAN.

Все приведенные примеры генерируются автоматически со следующими гиперпараметрами: размер батча 16 и 6, top-k 2048 и 768, top-p 0.995 и 0.99, температура 1.0, 1 GPU Tesla A100 для Malevich (а также Emojich XL) и Kandinsky (а также Emojich XXL) соответственно.

Таблица 3. Эмиссия CO_2 и энергопотребление для дообучения моделей Malevich и Kandinsky

Модель	Время обучения	Энергия, кВт · ч	CO_2, кг	GPU	CPU	Размер батча
Malevich	4h 19m	1.37	0.33	A100 Graphics, 1	AMD EPYC 7742 64-Core	4
Kandinsky	9h 45m	24.50	5.89	A100 Graphics, 8	AMD EPYC 7742 64-Core	12

Параметры дообучения, результаты потребления энергии и эквивалентной эмиссии CO_2 приведены в табл. 3. Можно отметить, что при дообучении Kandinsky выделяется более чем в 17 раз больше CO_2, чем при дообучении Malevich.

Таким образом, можно видеть, что библиотека *eco2AI* позволяет контролировать энергопотребление при обучении (и дообучении) больших моделей не только на одном GPU, но и на нескольких GPU, что имеет важное значение в случае использования библиотек оптимизации для распределенного обучения, например DeepSpeed.

4.2. Предобучение мультимодальных моделей

Обучение больших моделей, таких как Malevich, требует затраты большого количества ресурсов. В этом разделе рассматривается случай повышения энергоэффективности модели с использованием квантованной функции активации GELU. Квантованная GELU [29] – это разновидность функции активации GELU [21], которая сохраняет градиенты модели с разрешением в несколько бит, тем самым занимая меньше памяти GPU и затрачивая меньше вычислительных ресурсов (см. рис. 4). Если быть точнее, сравниваются ошибка и энергоэффективность версии модели Malevich с обычным GELU и версией Malevich с квантованным GELU 4-бит, 3-бит, 2-бит и 1-бит с помощью библиотеки *eco2AI*. Для всех версий моделей использова-лись тот же оптимизатор, планировщик и алгоритм обучения, что и в экспериментах по дообучению. Для обеспечения воспроизводимости каждый эксперимент запускался 5 раз со случайным начальным числом. Набор данных для обучения состоял из 300 000 объектов. Каждый образец был пропущен через модель только один раз с размером батча, равным 4. Набор данных для валидации состоял из 100 000 объектов. Для отслеживания углеродного следа во время обучения в режиме реального времени использовалась библиотека *eco2AI*.

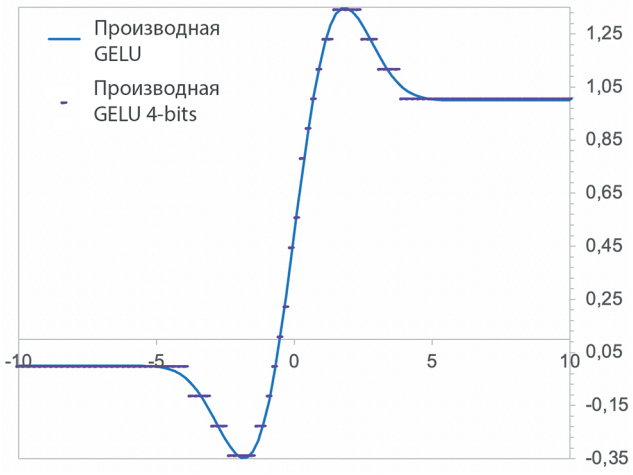

Рис. 4. Функция активации GELU и ее оптимизированная 4-битная аппроксимация.

Как видно из рис. 5а, потери при валидации Malevich с 4-битным, 3-битным GELU и Malevich с обычным GELU почти одинаковы (рост 0.06%), тогда как 2-битный, 1-битный GELU демонстрируют увеличение ошибки на валидационной выборке примерно на 0.42%. При этом, 4-битный GELU и 3-битный GELU примерно на 15 и 17% соответственно более энергоэффективны по сравнению с исходным GELU и приводят к соответственно меньшей эмиссии CO_2 на одном и том же шаге обучения (рис. 5б).

Использование 1-битного GELU позволило дополнительно уменьшить эмиссию только на 0.05% CO_2. Производительность моделей представлена на рис. 5в и в табл. 4. Таким образом, GELU 3-бит обеспечивает точность модели, близкую к оригинальной GELU, при этом потребляя на 17% меньше энергии и, следовательно, производя меньше эквивалентной эмиссии CO_2.

Таким образом, библиотека *eco2AI* может отслеживать энергопотребление и углеродный след при обучении моделей ИИ в режиме реального времени, помогает реализовывать и демонстрировать различные алгоритмы оптимизации энергопотребления (например, с помощью квантованых функций активации).

5. ЗАКЛЮЧЕНИЕ

Несмотря на большой потенциал ИИ в решении экологических проблем, сам ИИ может также быть источником углеродного следа. Библиотека *eco2AI* может помочь ИИ-сообществу понять влияние моделей ИИ на окружающую среду во время обучения и инференса и организовать систематический мониторинг эквивалентной эмиссии углерода. *eco2AI* – это библиотека с открытым

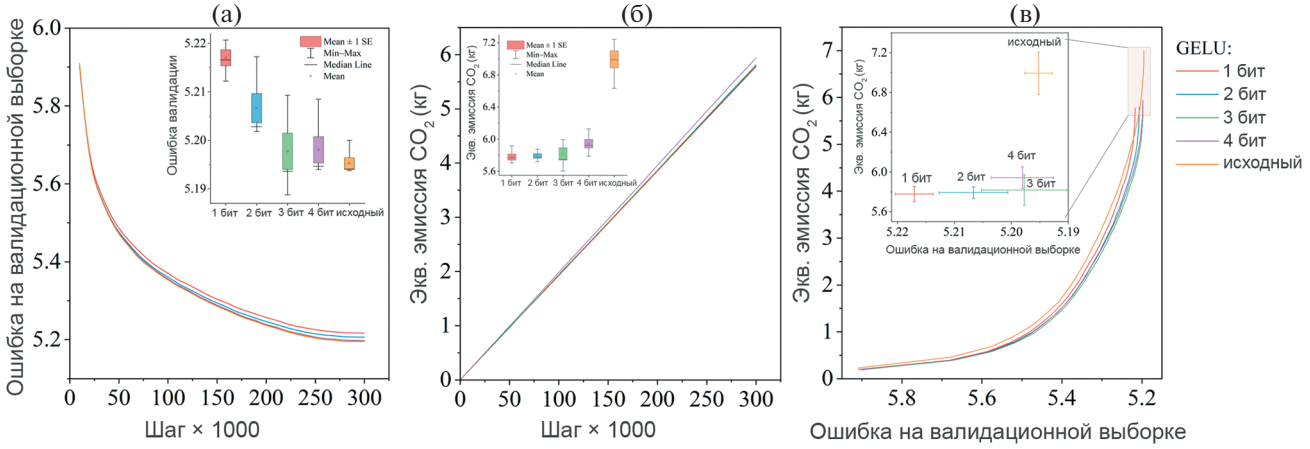

Рис. 5. Сравнение квантованных функций активации GELU и исходного GELU, использованных при обучении модели Malevich: (а) ошибка на валидационной выборке на каждом шаге обучения (ящичковая диаграмма на вставке показывает статистику ошибки при валидации каждой модели на 300000 шаге), (б) Эмиссия CO_2 на каждом шаге обучения моделей (диаграмма на вставке показывает статистику эмиссии CO_2 каждой модели на шаге 300000), (в) Эмиссия CO_2 для ошибки на валидационной выборке для каждой модели. (Для более наглядной демонстрации различия между моделями на вставке показана увеличенная область графика с пиковой эмиссией CO_2).

исходным кодом, рассчитывающая эквивалентную эмиссию углерода при обучении или инференсе моделей ИИ на Python исходя из энергопотребления GPU, CPU и RAM. В *eco2AI* делается акцент на точности расчета энергопотребления, что достигается с помощью учета региональных коэффициентов интенсивности эмиссии CO_2 и точным определением загрузки CPU.

В работе приведены примеры использования *eco2AI* для контроля эмиссии CO_2 при дообучении больших моделей text2image, таких как Malevich и Kandinsky, а также для оптимизации функции активации GELU, используемой при обучении модели Malevich. С помощью *eco2AI* было продемонстрировано, что использование 3-битного GELU позволяет уменьшить эквивалентную эмиссию CO_2 при обучении модели примерно на 17%. Авторы ожидают, что *eco2AI* может помочь сообществу перейти к “Зеленому ИИ”, в рамках предложенной концепции циклического снижения выбросов парниковых газов.

ПРИЛОЖЕНИЕ

Примеры использования библиотеки eco2AI

eco2AI – библиотека на Python с открытым исходным кодом, распространяется под лицензией Apache 2.0 [3]. Она доступна для установки с PyPI [32], а также ее исходный код можно найти на репозитории в GitHub [10]. Ниже представлены различные способы интеграции библиотеки *eco2AI* в код своего Python проекта.

После установки библиотеки, ее нужно импортировать в свой код, создать объект класса *eco2ai.Tracker()*, затем вызвать до основного кода метод *.start()* класса *eco2ai.Tracker()* и после него метод *.stop()* (см. Листинг 1).

Таблица 4. Эквивалентная эмиссия CO_2 и энергопотребление предобученной модели Malevich на наборе данных, состоящем из 300000 изображений за одну эпоху (A100 Graphics, AMD EPYC 7742 64-Core)

Модель	Время обучения, ч	Энергия, кВт · ч	CO_2, кг	Ошибка валидации
Malevich, GELU исходная	75.2 ± 1.3	29.1 ± 1	7.0 ± 0.24	5.195 ± 0.002
Malevich, GELU 4-бит	67.2 ± 1.5	24.7 ± 0.5	5.94 ± 0.12	5.198 ± 0.005
Malevich, GELU 3-бит	67.2 ± 1.5	24.2 ± 0.7	5.81 ± 0.17	5.198 ± 0.008
Malevich, GELU 2-бит	66.5 ± 0.3	24.0 ± 0.3	5.79 ± 0.06	5.207 ± 0.006
Malevich, GELU 1-бит	67.7 ± 0.73	24 ± 0.36	5.77 ± 0.09	5.217 ± 0.003

Листинг 1: Базовый пример использования

```
import eco2ai
tracker = eco2ai.Tracker(project_name="YourProjectName",
        experiment_description="training_the_<your_model>_model")

tracker.start()
<your gpu & (or) cpu calculations>
tracker.stop()
```

Еще одним способом встроить трекер в свой является использование декораторов (см. Листинг 2). Декоратор *track* нашей библиотеки позволяет модифицировать любую функцию так, чтобы при ее выполнении система вела расчет косвенно выделившегося CO_2.

Листинг 2: Пример использования декораторов

```
from eco2ai import track


@track
def train_func(model, dataset, optimizer, epochs, *args, **kwargs):
  ...
train_func(model, dataset, optimizer, epochs, *args, **kwargs)
```

После каждого вызова метода *.stop()* трекер завершит работу, и все результаты будут записаны в локальный файл *“emission.csv”*, название которого также можно задавать параметром *file_name* класса *eco2ai.Tracker()*. Результирующий файл представляет собой таблицу со следующими столбцами:

- *id* – уникальный id эксперимента;
- *project_name* – название проекта, задаваемое пользователем;
- *experiment_description* – описание эксперимента, также задаваемое пользователем;
- *epoch* – информация об эпохе обучения, в случае, если используется данная функция;
- *start_time* – дата начала эксперимента в формате yyyy-mm-dd hh:mm:ss;
- *duration(s)* – длительность эксперимента в секундах;
- *power_consumption(kWh)* – энергия, косвенно выделившаяся за время эксперимента, выраженная в кВт·ч;
- *CO_2_emissions(kg)* – масса косвенно выделившегося углекислого газа за время эксперимента, выраженная в кг;
- *CPU_name* – информация о наименовании и количестве CPU;
- *GPU_name* – информация о наименовании и количестве GPU;
- *OS* – название ОС устройства, на котором проводился эксперимент;
- *region/country* – регион и страна, определяющиеся автоматически по IP или задаваемые пользователем вручную с помощью параметров *alpha_2_code* и *region* класса *eco2ai.Tracker()*;
- *cost* – стоимость затраченной электроэнергии в условных единицах. Расчитывается в случае, если пользователь задал параметр *electricity_pricing*;

eco2AI позволяет пользователю записывать информацию об обучении в зашифрованном виде (см. Листинг 3). Эта функция может быть полезна в тех случаях, когда требуется защитить файл с результатами от ручной модификации пользователем. Для кодировки результатов необходимо задать параметр *“encode_file”* класса *eco2ai.Tracker()*, тогда в файл эксперимента будут записаны защифрованные данные *“encoded_emission.csv”*.

Листинг 3: Функция кодирования результатов

```
import eco2ai

tracker = eco2ai.Tracker(
        file_name='encoded_emissions.csv',
        project_name="Test_1",
        experiment_description="testing_Eco2AI_in_encoding_mode",
        encode_file=True,
)
```

В *eco2AI* также реализована возможность контролировать эквивалентную эмиссию CO_2 в каждой эпохе обучения моделей машинного и глубокого обучения. В Листинге 4 разобран простейший пример использования данной возможности.

Листинг 4: Эксперимент для глубокого обучения

```
tracker = eco2ai.Tracker(
        project_name="CIFAR10_for_ESG",
        experiment_description="ML_tracking",
        file_name="emission.csv",
        emission_level=None,
        alpha_2_code="AU",
        region="Queensland",
        )
tracker.start_training()

net = Net()
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9)
parameters_to_save = dict()
for epoch in range(epochs):
parameters_to_save["loss"] = train_epoch(net, optimizer, trainloader)
        parameters_to_save["train_accuracy"] = get_accuracy(net, trainloader)
        parameters_to_save["test_accuracy"] = get_accuracy(net, testloader)

        tracker.new_epoch(parameters_to_save)
tracker.stop_training()
```

Подробную документацию для каждого метода класса *eco2ai.Tracker()* и его параметров можно вызвать с помощью функции *help(eco2ai.Tracker())*.

СПИСОК ЛИТЕРАТУРЫ

1. Paris Agreement. Paris agreement. In Report of the Conference of the Parties to the United Nations Framework Convention on Climate Change (21st Session, 2015: Paris). Retrived December, volume 4, page 2017. HeinOnline, 2015. Open AI. URL https://openai.com/blog/ai-and-compute/
2. *Lasse F Wolff Anthony, Benjamin Kanding, and Raghavendra Selvan.* Carbontracker: Tracking and predicting the carbon footprint of training deep learning models. arXiv preprint arXiv:2007.03051, 2020.
3. Apache licence 2.0. URL https://www.apache.org/licenses/LICENSE-2.0.
4. Carbontracker. URL https://github.com/lfwa/carbontracker.
5. Cloud Carbon Footprint. URL https://github.com/cloud-carbon-footprint/cloud-carbon-footprint.
6. Codecarbon. URL https://github.com/mlco2/codecarbon.
7. *Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer.* 8-bit optimizers via blockwise quantization. arXiv preprint arXiv:2110.02861, 2021.
8. *Payal Dhar.* The carbon impact of artificial intelligence. Nat. Mach. Intell., 2(8):423–425, 2020.
9. *Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith.* Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. arXiv preprint arXiv:2002.06305, 2020.
10. eco2ai github. URL https://github.com/sb-ai-lab/eco2AI.
11. Ember. Global electricity review 2022, Mar 2022. URL https://ember-climate.org/insights/research/global-electricity-review-2022/.
12. EMISS. The unified interdepartmental information and statistical systems (emiss). URL https://fedstat.ru/indicator/58506.
13. Emissions factors sources, 2021. URL https://www.carbonfootprint.com/docs/2022_01_emissions_factors_-sources_for_2021_electricity_v10.pdf.
14. emojich ruDALL-E. URL https://www.kaggle.com/shonenkov/emojich-rudall-e.

15. *Patrick Esser, Robin Rombach, and Björn Ommer.* Taming transformers for high-resolution image synthesis, 2020.
16. Experiment impact tracker. URL https://github.com/Breakend/experiment-impact-tracker.
17. Green algorithms tool. URL https://github.com/GreenAlgorithms/green-algorithms-tool.
18. *Julia Gusak, Daria Cherniuk, Alena Shilova, Alexander Katrutsa, Daniel Bershatsky, Xunyi Zhao, Lionel Eyraud-Dubois, Oleg Shlyazhko, Denis Dimitrov, Ivan Oseledets, and Olivier Beaumont.* Survey on large scale neural network training, 2022.
19. *Song Han, Huizi Mao, and William J Dally.* Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149, 2015.
20. *Peter Henderson, Jieru Hu, Joshua Romoff, Emma Brunskill, Dan Jurafsky, and Joelle Pineau*. Towards the systematic reporting of the energy and carbon footprints of machine learning. Journal of Machine Learning Research, 21(248):1–43, 2020.
21. *Dan Hendrycks and Kevin Gimpel.* Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415, 2016.
22. *Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al.* Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2(7), 2015.
23. *Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi*. The curious case of neural text degeneration. CoRR, abs/1904.09751, 2019. URL http://arxiv.org/abs/1904.09751.
24. *Vladan Joler Kate Crawford*. Anatomy of an ai system, 2018. URL http://www.anatomyof.ai.
25. *Matt J. Kusner and José Miguel Hernández-Lobato*. Gans for sequences of discrete elements with the gumbel-softmax distribution, 2016.
26. *Loc Lannelongue, Jason Grealey, and Michael Inouye.* Green algorithms: quantifying the carbon footprint of computation. Advanced science, 8(12):2100707, 2021.
27. *DA Maevsky, EJ Maevskaya, and ED Stetsuyk*. Evaluating the ram energy consumption at the stage of software development. In Green IT Engineering: Concepts, Models, Complex Systems Architectures, pages 101–121. Springer, 2017.
28. Minprirody (Russia). URL https://xn–d1ahaoghbejbc5k.xn–p1ai/documents/active/664/.
29. *Georgii Novikov, Daniel Bershatsky, Julia Gusak, Alex Shonenkov, Denis Dimitrov, and Ivan Oseledets.* Few-bit backward: Quantized gradients of activation functions for memory footprint reduction, 2022.
30. *David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean*. Carbon emissions and large neural network training. arXiv preprint arXiv:2104.10350, 2021.
31. *Pesce M*. Cloud computing's coming energy crisis. IEEE Spectrum, 2021.
32. PyPi Eco2AI. URL https://pypi.org/project/eco2AI/.
33. *Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever.* Zero-shot text-to-image generation. In Marina Meila and Tong Zhang, editors, Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event, volume 139 of Proceedings of Machine Learning Research, pages 8821–8831. PMLR, 2021. URL http://proceedings.mlr.press/v139/ramesh21a.html.
34. Rosstat. Russian federal state statistics service. URL https://rosstat.gov.ru/enterprise_industrial.
35. russian-emoji. URL https://www.kaggle.com/datasets/shonenkov/russian-emoji.
36. Science and Resources. National greenhouse accounts factors. URL https: //www.industry.gov.au/sites/default/files/August%202021/document/national-greenhouse-accounts-factors-2021.pdf.
37. *Alex Shonenkov, Daria Bakshandaeva, Denis Dimitrov, and Aleksandr Nikolich*. Emojich zero-shot emoji generation using russian language: a technical report. arXiv preprint arXiv:2112.02448, 2021.
38. *Emma Strubell, Ananya Ganesh, and Andrew McCallum.* Energy and policy considerations for deep learning in nlp. arXiv preprint arXiv:1906.02243, 2019.
39. Tracarbon. URL https://github.com/fvaleye/tracarbon.
40. UNFCCC. Canada, national inventory report, 2021. URL https://unfccc.int/sites/default/files/resource/can-2021-nir-12apr21.zip. Part 3, page 60.
41. USA EPA. egrid2020, May 2020. URL https://www.epa.gov/system/files/documents/2022-01/egrid2020_data.xlsx.
42. *Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin.* Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M.Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 5998–6008, 2017. URL https://proceedings.neurips. cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.
43. *Ricardo Vinuesa, Hossein Azizpour, Iolanda Leite, Madeline Balaam, Virginia Dignum, Sami Domisch, Anna Felländer, Simone Daniela Langhans, Max Tegmark, and Francesco Fuso Nerini*. The role of artificial intelligence in achieving the sustainable development goals. Nature communications, 11(1):1–10, 2020.
44. *YouTokenToMe*. URL https://github.com/VKCOM/YouTokenToMe.
45. ZeRO-3. URL https://www.deepspeed.ai/2021/03/07/zero3-offload.html.