MLP-Mixer: An all-MLP Architecture for Vision

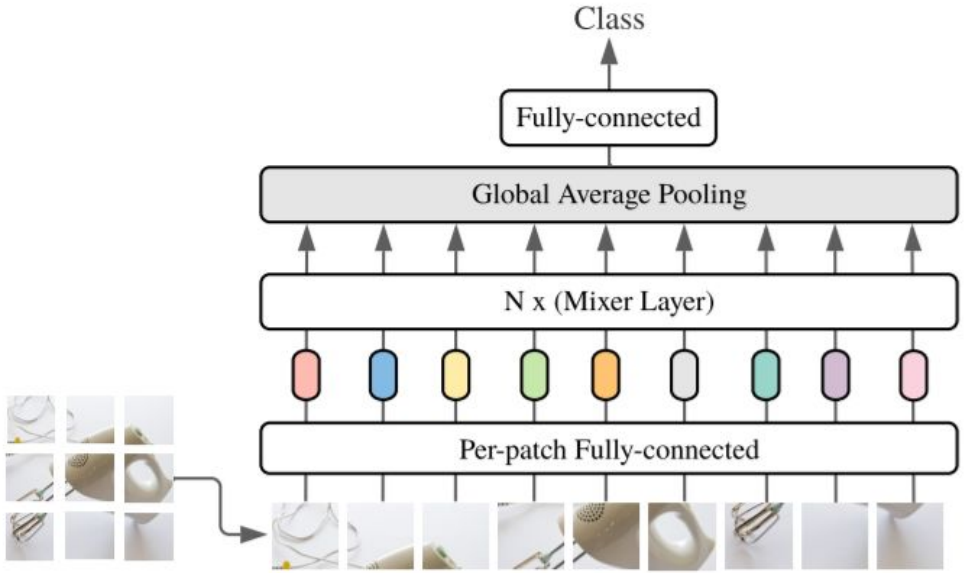
Докладчик: Дарья Виноградова
Рецензент: Александра Коган
Практик-исследователь: Иван Сафонов
Хакер: Рамазан Рахматуллин

Мотивация

- Все используют CNN и трансформеры для хороших результатов в задачах CV
- А что если отойти от стандарта?
- Предложение: использовать в архитектуре только MLP
- Получится ли модель конкурентной для де-факто стандартных CNN и трансформеров?

Архитектура

Общая идея: модель вычисляет признаки в данном пространственном блоке, а затем замешивает признаки со всех блоков



4. Применяем Global Average Pooling и классифицируем

3. С помощью MLP смешиваем патчи

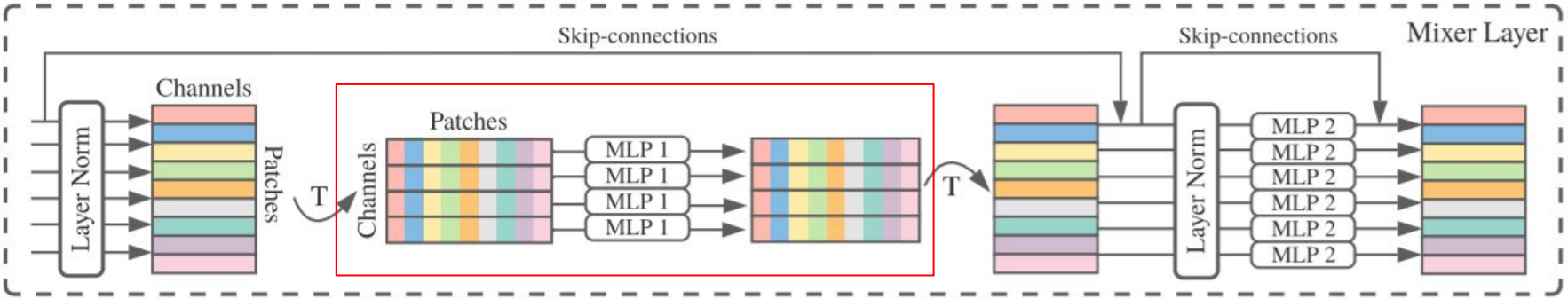
2. Применяем MLP для каждого патча

1. Делим изображение на патчи

Архитектура: проектор

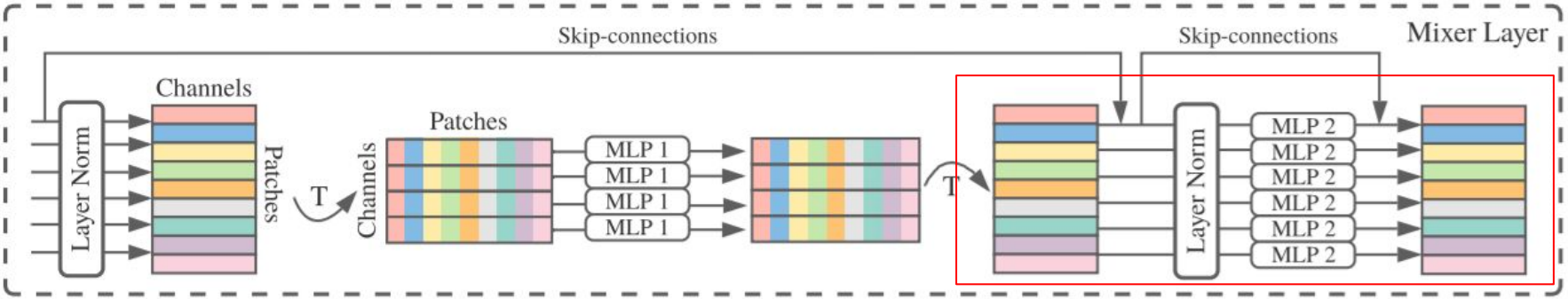
- Изображение делится на патчи размером PxP (*S=(WH)/P²* штук)
- Каждый патч вытягивается в вектор длины P²
- К нему применяется линейный проектор
- На выходе - вектор длины C (*C-канальный*)

Архитектура: миксер



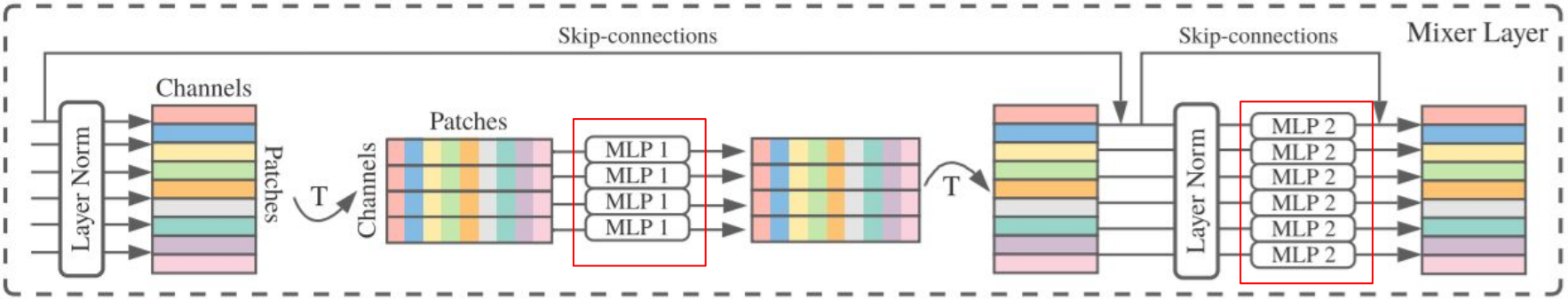
1. MLP применяется *поканально* по всем патчам (по столбцам) $\mathbb{R}^C \mapsto \mathbb{R}^C$,

Архитектура: миксер



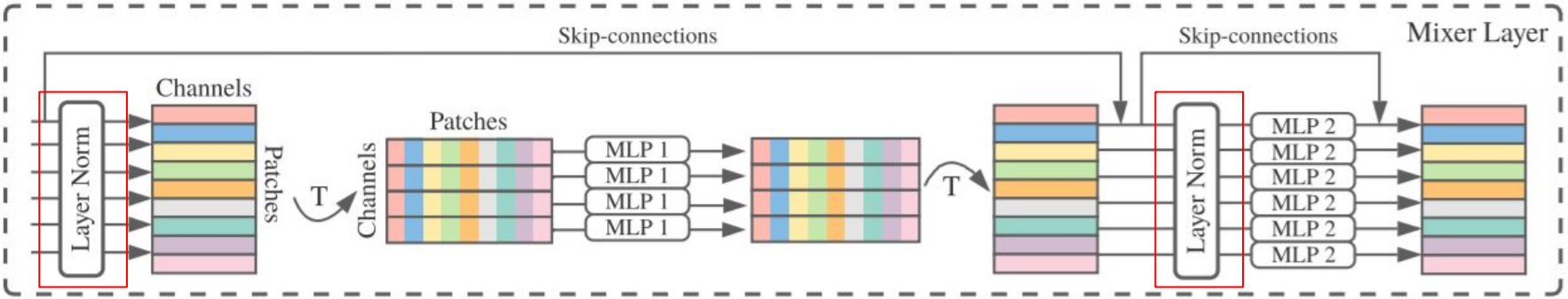
1. MLP применяется *поканально* по всем патчам (по столбцам) $\mathbb{R}^C \mapsto \mathbb{R}^C$,
2. MLP применяется к каждому патчу *отдельно* (по строкам) $\mathbb{R}^S \mapsto \mathbb{R}^S$

Архитектура: миксер



1. MLP применяется *поканально* по всем патчам (по столбцам) $\mathbb{R}^C \mapsto \mathbb{R}^C$,
2. MLP применяется к каждому патчу *отдельно* (по строкам) $\mathbb{R}^S \mapsto \mathbb{R}^S$
3. MLP = Linear + GeLU + Linear

Архитектура: миксер



1. MLP применяется *поканально* по всем патчам (по столбцам) $\mathbb{R}^C \mapsto \mathbb{R}^C$,
2. MLP применяется к каждому патчу *отдельно* (по строкам) $\mathbb{R}^S \mapsto \mathbb{R}^S$
3. MLP = Linear + GeLU + Linear
4. LayerNorm перед применением MLP

Архитектура: сравнение с CNN и трансформерами

	Смешивание признаков внутри одного блока	Смешивание признаков из разных блоков
CNN	1x1 свертка	Фильтр, захватывающий всю картинку поканально
Transformer	Self-attention	
Mixer	MLP по каналам	MLP по патчам

CNN и трансформеры не разделяют две задачи. Миксер - разделяет

Варианты моделей

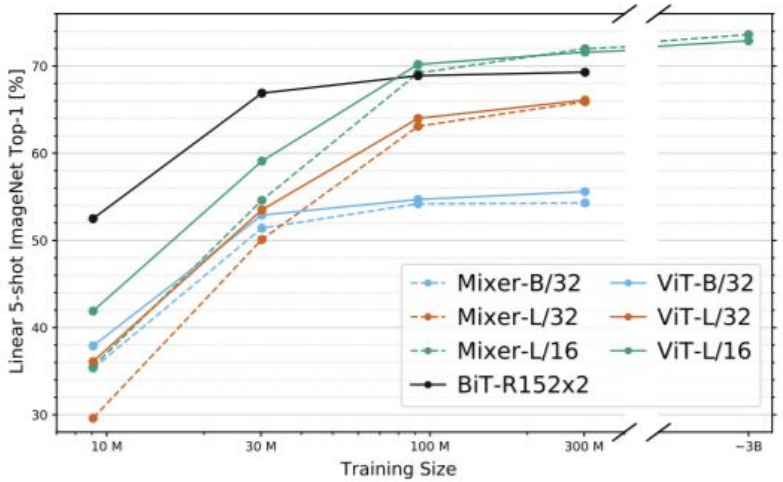
Specification	S/32	S/16	B/32	B/16	L/32	L/16	H/14
Number of layers	8	8	12	12	24	24	32
Patch resolution $P \times P$	32×32	16×16	32×32	16×16	32×32	16×16	14×14
Hidden size C	512	512	768	768	1024	1024	1280
Sequence length S	49	196	49	196	49	196	256
MLP dimension D_C	2048	2048	3072	3072	4096	4096	5120
MLP dimension D_S	256	256	384	384	512	512	640
Parameters (M)	19	18	60	59	206	207	431

Эксперименты: ImageNet

	ImNet top-1	ReaL top-1	Avg 5 top-1	VTAB-1k 19 tasks	Throughput img/sec/core	TPUv3 core-days
Pre-trained on ImageNet-21k (public)						
HaloNet [51]	85.8	—	—	—	120	0.10k
Mixer-L/16	84.15	87.86	93.91	74.95	105	0.41k
ViT-L/16 [14]	85.30	88.62	94.39	72.72	32	0.18k
BiT-R152x4 [22]	85.39	—	94.04	70.64	26	0.94k
Pre-trained on JFT-300M (proprietary)						
NFNet-F4+ [7]	89.2	—	—	—	46	1.86k
Mixer-H/14	87.94	90.18	95.71	75.33	40	1.01k
BiT-R152x4 [22]	87.54	90.54	95.33	76.29	26	9.90k
ViT-H/14 [14]	88.55	90.72	95.97	77.63	15	2.30k
Pre-trained on unlabelled or weakly labelled data (proprietary)						
MPL [34]	90.0	91.12	—	—	—	20.48k
ALIGN [21]	88.64	—	—	79.99	15	14.82k

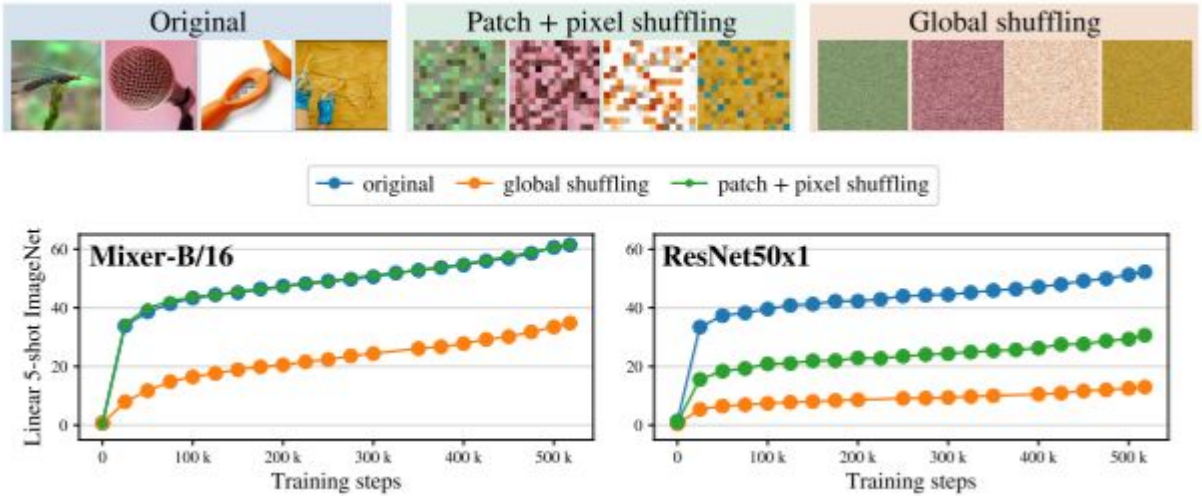
Эксперименты: Linear 5-shot Imagemet

При большом датасете для предобучения Mixer лишь немного отстает от state-of-the-art моделей



Linear 5-shot со схемы - классификация на 5 категорий предобученной моделью с помощью только линейного слоя в конце

Эксперименты: устойчивость к перестановкам



Выводы и перспективы

- Придумана новая архитектура, лишь немного отстающая от state-of-the-art моделей в задаче классификации
- Очень хорошо показывает себя при предобучении на большом датасете
- Обладает определенной устойчивостью к попиксельным перестановкам
- Вероятно может находить новые признаки, которые не видят стандартные архитектуры
- Достойна применения в задачах из других областей

Рецензия

Содержание:

Простая архитектура для CV, без self-attention, только с использованием MLP

Вклад:

Первые мощные результаты с MLP в CV

Сильные стороны:

- Хорошо написана
- Визуализации
- Есть несколько спецификаций, несколько вариантов предобучения
- Воспроизводимость (Есть код на языке Jax)

Слабые стороны:

- Хорошие результаты только на больших наборах данных
- Нет сравнения с state-of-the-art моделями:
  - BiT - декабрь 2019 года (сравнивается с)
  - ViT - октыбрь 2020 года (сравнивается с)
  - EfficientNetV2 - апрель 2021 (не сравнивается)
  - MLP-Mixer - май 2021
- Нет FLOPs (удобно для сравнения стоимости обучения)

Оценка

Оценка по критериям НИПСа: 7

Уверенность в оценке: 4

Хорошие результаты только на больших наборах данных

История

- Работа написана весной 2021 года.
- Опубликована на arXiv 4 мая 2021 (первая версия), 10 июня 2021 (четвертая версия); на OpenReview 21 мая 2021 года.
- Была постером на NeurIPS 2021.

Авторы

- Авторы статьи: Google Research, Brain Team

- **Ilya Tolstikhin**: CV, GAN.
  Most cited: «Wasserstein auto-encoders»
- **Neil Houlsby**: CV, NLP, Bayesian methods.
  Most cited: «An image is worth 16x16 words: Transformers for image recognition at scale»
- **Alexander Kolesnikov**: CV.
  Most cited: «An image is worth 16x16 words: Transformers for image recognition at scale»
- **Lucas Beyer**: RL, CV
  Most cited: «In Defense of the Triplet Loss for Person Re-Identification», «Revisiting Self-Supervised Visual Representation Learning»

Авторы

- Xiaohua Zhai: RL, CV.

  Most cited: «An image is worth 16x16 words: Transformers for image recognition at scale», «Revisiting Self-Supervised Visual Representation Learning»
- Thomas Unterthiner: CV, GAN, Bioinformatics.

  Most cited: «Fast and accurate deep network learning by exponential linear units (ELUs)», «GANs trained by a two time-scale update rule converge to a local nash equilibrium», «Self-normalizing neural networks»
- Jessica Yung: CV (transfer learning, representation learning)

  Most cited: «Big Transfer (BiT): General Visual Representation Learning»
- Andreas Peter Steiner: CV, Bioinformatics.

  Most cited: «MLP-Mixer: An all-MLP Architecture for Vision»
- Daniel Keysers: CV

  Выпускал работы начиная с 2001, много популярных старых работ.
- Jakob Uszkoreit: CV, NLP

  Most cited: соавтор «Attention is all you need»
- Mario Lucic: CV, GAN

  Most cited: «Are GANs Created Equal? A Large-Scale Study»
- Alexey Dosovitskiy: CV

  Один из основных авторов Visual Transformer (статья «An image is worth 16x16 words: Transformers for image recognition at scale»)

Ссылки

- Всего 60 ссылок
- Наибольшее влияние: Visual Transformer, статья «An image is worth 16x16 words: Transformers for image recognition at scale»

Цитирования

- Пока есть 114 цитирований

Наиболее интересные:

- «Do Vision Transformers See Like Convolutional Neural Networks?»
- «Pay Attention to MLPs»
- «A Battle of Network Structures: An Empirical Study of CNN, Transformer, and MLP»

Идеи дальнейших исследований
Объединение архитектур

- Давайте вместо обучаемой линейной проекции патча $B \times B$ изображения использовать CNN, в конце которой получится вектор из C каналов, который уже будет передаваться в MLP Layers. Интересно посмотреть на разные значения B. Также возможно это поможет легко решить проблему применимости модели к картинкам разного размера (потому что сначала применяется CNN, которая может быть применена к любым размерам).
- Что будет, если в модели использовать и слои транформера и Mixer Layers? Несколько возможных вариаций: по очереди в некотором порядке, либо каждый слой это сумма слоя транформера и Mixer Layer.
- Размещать CNN блоки можно не только в начале, также можно: после любого слоя модели представить, что у нас изображение размера $\frac{H}{B} \times \frac{W}{B}$ с C каналами, применять CNN слой.

Идеи дальнейших исследований
Другие задачи CV

Еще одна возможная идея: применение MLP-Mixer к задаче сегментации. Можно получившиеся в конце S векторов размера C также обучаемой линейной проекцией перевести в блоки размера $B \times B$ и соединить в картинку исходного размера.

Идеи практического применения

Авторы показывают, что MLP-Mixer обучается эффективнее на TPU, чем другие модели.

Возможно, у нее также более эффективный inference. Возможно можно немного пожертвовав в качестве ускорить время применения в задачах классификации на CPU/мобильных устройствах.