Национальный исследовательский
Нижегородский государственный университет им. Н.И. Лобачевского
Институт информационных технологий, математики и механики

Образовательный курс
«Введение в глубокое обучение с использованием Intel® neon™ Framework»

Перенос обучения глубоких нейронных сетей

При поддержке компании Intel

Кустикова Валентина,
к.т.н., ст.преп. каф. МОСТ ИИТММ,
ННГУ им. Н.И. Лобачевского

Содержание

- Понятие переноса обучения
- Практическое применение переноса обучения
- Перенос обучения в Intel® neon™ Framework

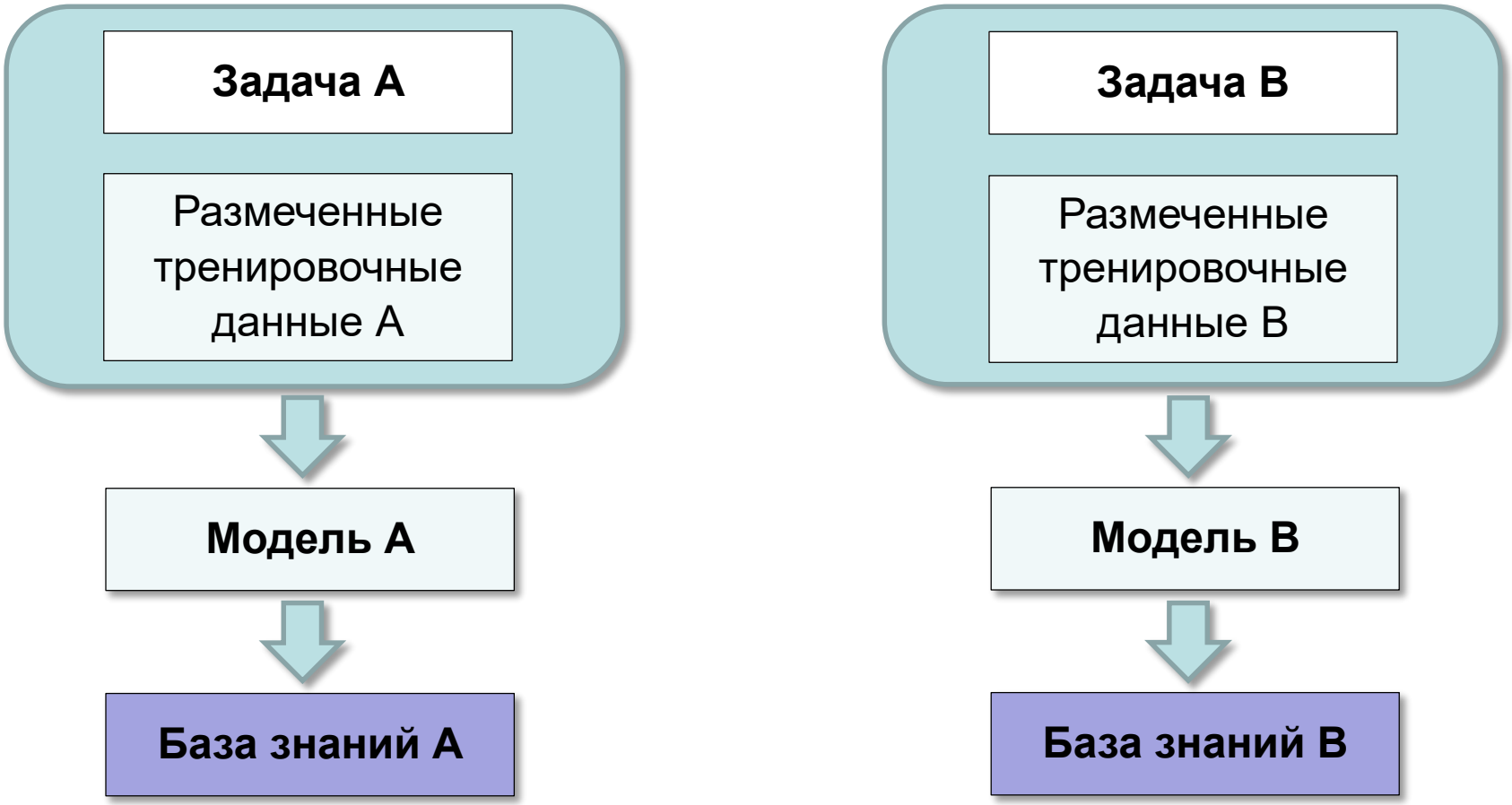
ПОНЯТИЕ ПЕРЕНОСА ОБУЧЕНИЯ

Введение (1)

- В настоящее время исследователи научились строить эффективные глубокие модели для решения большого спектра задач компьютерного зрения, обработки изображений и естественного языка
- Обучение подобных моделей требует значительного объема размеченных тренировочных данных, поскольку большинство задач решается с использованием методов обучения с учителем

Введение (2)

- «Задача A» – исходная задача
- «Задачи B» – целевая задача



Введение (3)

- Если исходная «Задача A» и целевая «Задача B» связаны некоторым образом, можно ли как-то использовать знания, полученные при решении исходной задачи?

Введение (4)

- Если исходная «Задача А» и целевая «Задача В» связаны некоторым образом, можно ли как-то использовать знания, полученные при решении исходной задачи?

- Да, но как?

Введение (5)

- Если исходная «Задача А» и целевая «Задача В» связаны некоторым образом, можно ли как-то использовать знания, полученные при решении исходной задачи?

- Да, но как?
- Пример:
  - Исходная «Задача А» – детектирование пешеходов в дневное время суток
  - Целевая «Задача В» – детектирование пешеходов в ночное время суток
  - Естественно использовать модель, обученную для решения исходной задачи, для решения целевой задачи

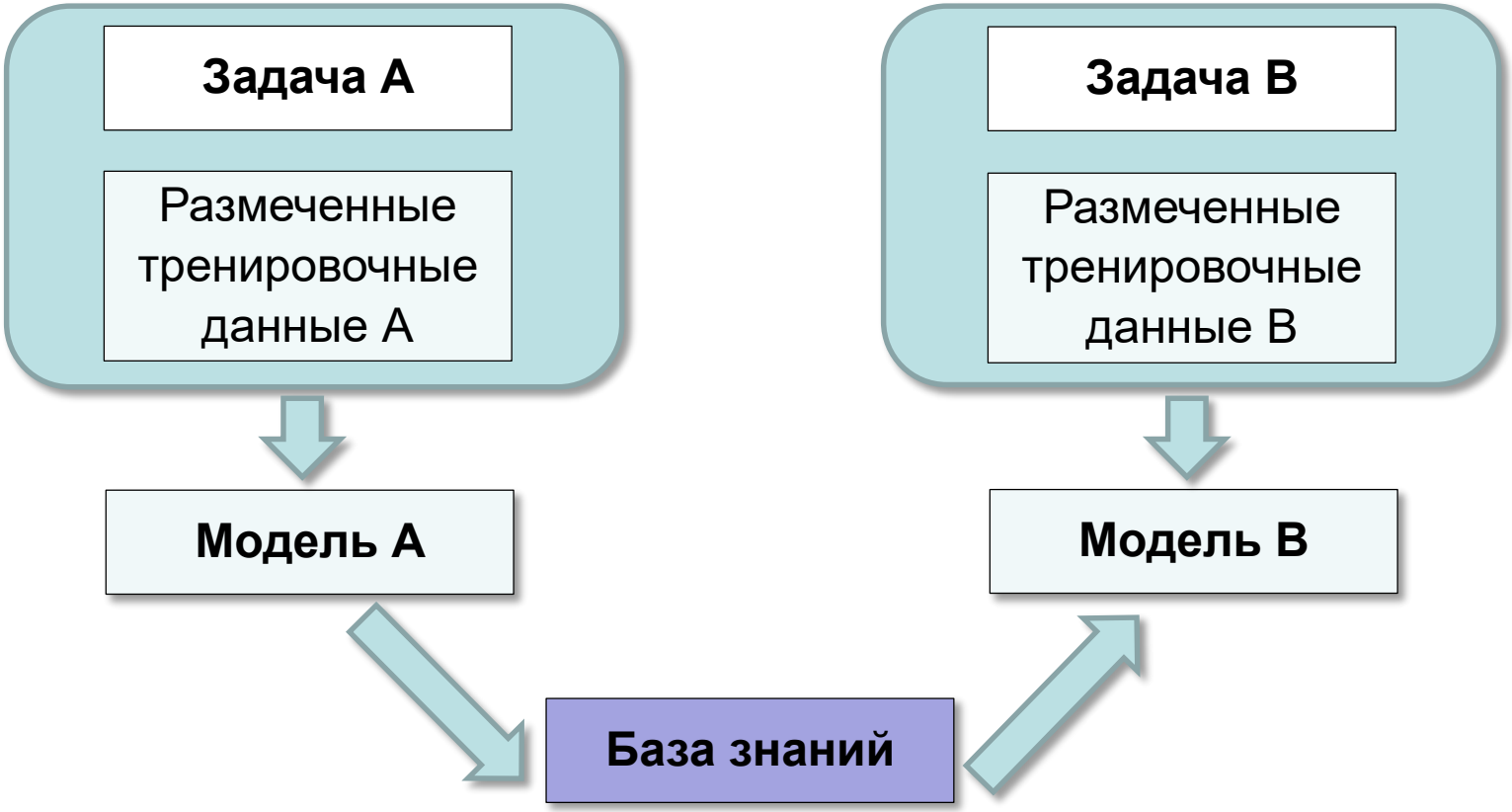
Введение (6)

- Если исходная «Задача А» и целевая «Задача В» связаны некоторым образом, можно ли как-то использовать знания, полученные при решении исходной задачи?

- Да, но как?
- Пример:
  - Исходная «Задача А» – детектирование пешеходов в дневное время суток
  - Целевая «Задача В» – детектирование велосипедистов
  - Исходную модель напрямую нельзя использовать для решения целевой задачи
  - Несмотря на то, что в обоих случаях осуществляется поиск людей, положение человека разное

Цель переноса обучения

- ***Цель переноса обучения*** – накопить знания, необходимые для решения исходной «Задачи А», и использовать их для решения близкой по смыслу целевой «Задачи В»



Формализация цели переноса обучения. Домен

- ***Домен*** определяется парой

$$D = \{\mathrm{X}, P(\mathrm{x})\},$$

где X – пространство признаков,

$P(\mathrm{x})$ – распределение вероятности случайной величины

$$\mathrm{x} = \{x_1, x_2, \dots, x_n\} \in \mathrm{X}$$

- Два домена являются несовпадающими $D_1 \neq D_2$, если различаются пространства признаков $\mathrm{X}_1 \neq \mathrm{X}_2$, либо распределения $P(\mathrm{x}_1) \neq P(\mathrm{x}_2)$

Формализация цели переноса обучения. Задача

- ***Задача*** в свою очередь определяется парой

$$T = \{\mathrm{Y}, f(\cdot)\},$$

где Y – пространство меток, $f(\cdot)$ – функция предсказания

- Задача не наблюдается в явном виде, она обучается на тренировочном множестве пар элементов $(x_i, y_i), x_i \in \mathrm{X}, y_i \in \mathrm{Y}$
- Функция предсказания представляет собой распределение условной вероятности того, что при заданном признаковом описании наблюдается определенная метка $f(x) = P(y|x)$
- Две задачи являются несовпадающими $T_1 \neq T_2$, если различаются пространства меток $\mathrm{Y}_1 \neq \mathrm{Y}_2$, либо распределения вероятностей $P(Y_1|\mathrm{x}_1) \neq P(Y_2|\mathrm{x}_2)$

Формализация цели переноса обучения

- При заданной исходной задаче T_S в домене D_S и целевой задаче T_T в домене D_T ***цель переноса обучения*** состоит в повышении качества функции предсказания $f_T(\cdot)$ целевой задачи в домене D_T с использованием знаний, полученных при обучении задачи T_S в домене D_S, где $D_T \neq D_S$ или $T_T \neq T_s$ (домены или задачи не совпадают)

Основные задачи переноса обучения

- ***«Что переносить?»***
  - Предполагает выделение части задачи или домена, которая должна быть перенесена
  - Существуют общие знания для доменов, справедливые для множества доменов одновременно, а есть специфичные для домена знания
- ***«Как переносить?»***
  - При наличии информации о том, «что переносить», вопрос «как переносить?» решается естественным образом
- ***«Когда переносить?»***
  - Ответ позволяет оценить ситуации, в которых применение переноса обучения является обоснованным
  - Если домены не имеют отношения друг к другу, то перенос обучения может быть неэффективен

Виды окружения при реализации переноса обучения (1)

- ***Индуктивный перенос обучения*** (Inductive Transfer Learning) – задачи не совпадают $T_T \neq T_S$
- При этом домены могут быть как одинаковыми, так и могут различаться
- Для создания объективной функции предсказания необходимы размеченные данные в рамках целевого домена

Виды окружения при реализации переноса обучения (2)

- Разделение индуктивного переноса обучения на два класса:
  - Наличие размеченных тренировочных данных в исходной задаче
    - Индуктивный перенос обучения = ***многозадачное обучение*** (multitask learning)
    - Разница состоит в цели переноса – повысить производительность модели при решении целевой задачи, а цель многозадачного обучения – обучить исходную и целевую задачу одновременно
  - Отсутствие размеченных тренировочных данных в исходной задаче
    - Индуктивный перенос обучения = ***самообучение*** (self-taught learning)
    - Пространства меток в доменах исходной и целевой задачи могут быть различны, что подразумевает отсутствие возможности прямого использования побочной информации исходного домена

Виды окружения при реализации переноса обучения (3)

- ***Трансдуктивный перенос обучения*** (Transductive Transfer Learning) – задачи совпадают $T_T = T_S$, а домены в которых они заданы, различаются $D_T \neq D_S$
- В данном случае отсутствуют размеченные тренировочные данные в целевом домене, но имеется большой объем размеченных тренировочных данных в исходном домене
- Возможны две ситуации:
  - Пространства признаков исходного и целевого домена различны $\mathrm{X}_S \neq \mathrm{X}_T$
  - Пространства признаков исходного и целевого домена одинаковы $\mathrm{X}_S = \mathrm{X}_T$, но распределения вероятности $P(\mathrm{X}_S) \neq P(\mathrm{X}_T)$ различаются

Виды окружения при реализации переноса обучения (4)

- ***Перенос обучения без учителя*** (Unsupervised Transfer Learning) – задачи не совпадают $T_T \neq T_S$, но цель состоит в решении задач обучения без учителя (кластеризация, снижение размерности пространства признаков и др.) в рамках домена целевой задачи D_T

Типы переноса обучения (1)

- ***Перенос обучения на основе экземпляров*** (instance-based transfer learning)
  - Используются некоторые части тренировочного множества из исходного домена в ходе обучения целевой задачи
- ***Перенос признакового представления*** (feature representation transfer)
  - Цель данного типа переноса состоит в том, чтобы обучить «хорошее» представление признаков целевого домена
  - Знания, используемые для передачи между доменами, кодируются в представлении обучаемой функции

Типы переноса обучения (2)

- ***Перенос параметров*** (parameter transfer)
  - Исходная и целевая задачи разделяют некоторые параметры или априорные распределения гиперпараметров моделей
  - Знания передаются через задачи
- ***Перенос обучения для родственных доменов*** (transfer learning for relational domains)
  - Некоторые отношения между данными в исходном и целевом доменах подобны
  - В этом случае передаваемые знания – отношения между данными

Применение различных типов переноса обучения в разном окружении

	Индуктивный перенос обучения $T_T \neq T_S$	Трансдуктивный перенос обучения $T_T = T_S, D_T \neq D_S$	Перенос обучения без учителя $T_T \neq T_S$
Перенос обучения на основе экземпляров	+	+	
Перенос признакового представления	+	+	+
Перенос параметров	+		
Перенос обучения для родственных доменов	+		

* Pan S.J., Yang Q. A Survey on Transfer Learning // IEEE Transactions on Knowledge and Data Engineering. – 2010. – Vol. 22, Issue 10. – P.1345-1359.

ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ ПЕРЕНОСА ОБУЧЕНИЯ

Практическое применение переноса обучения

- Практическое применения переноса обучения при решении большого количества задач обучения с учителем сводится к проведению различного типа экспериментов

- Постановка задачи:
  - Допустим, что обучена глубокая модель для решения исходной «Задачи А»
  - Необходимо получить модель для решения целевой «Задачи В». Предполагается, что задачи связаны по смыслу

Примеры

- «Задача А» – классификация изображений, а «Задача В» – классификация изображений автомобилей
- «Задача А» – детектирование объектов разных классов, а «Задача В» – детектирование транспортных средств разных классов

Виды экспериментов (1)

- ***Прямое использование модели, обученной для решения исходной «Задачи А», для решения целевой «Задачи В»***
  - Данная схема позволяет оценить качество работы модели, построенной для решения общей задачи, применительно к частной задаче
  - Обученная глубокая модель используется без изменений
  - Эксперимент реализует перенос параметров

Виды экспериментов (2)

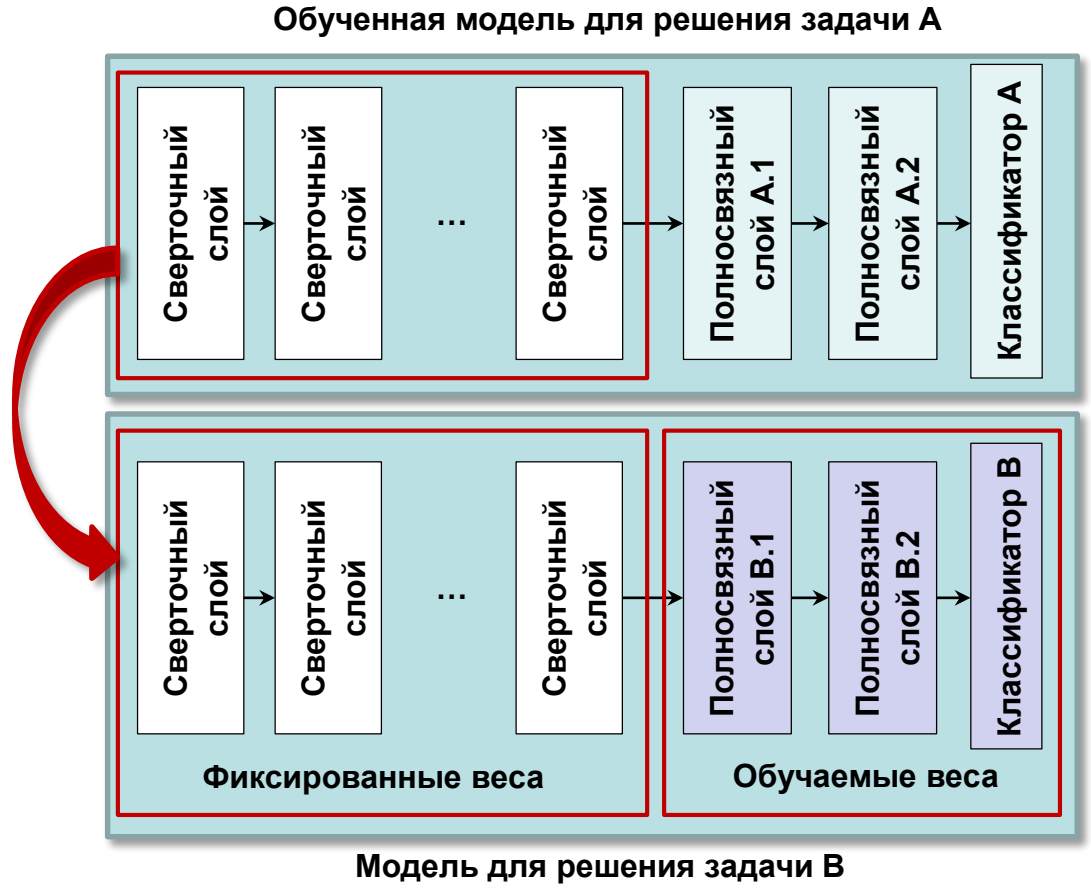
- ***Использование структуры глубокой модели, построенной для решения исходной «Задачи А», с целью обучения аналогичной модели для решения «Задачи В»***
  - Предполагается, что модель, построенная для решения исходной задачи, обучается на данных, подготовленных для решения целевой задачи
  - При этом веса модели инициализируются случайным образом
  - Эксперимент реализует перенос знаний для родственных доменов

Виды экспериментов (3.1)

- ***Использование модели, построенной для решения исходной «Задачи А», в качестве фиксированного метода извлечения признаков при построении модели, решающей «Задачу В»***
  - Идея данного подхода состоит в том, чтобы удалить из глубокой модели классификатор (последние полностью связанные слои) и рассматривать начальную часть сети как метод выделения признаков
  - Взамен старого классификатора можно поместить новый классификатор (например, другой набор полностью связанных слоев или машину опорных векторов) и обучить его на признаках, построенных с использованием начальной части сети
  - Эксперимент реализует перенос признакового описания

Виды экспериментов (3.2)

- ***Использование модели, построенной для решения исходной «Задачи А», в качестве метода извлечения признаков при построении модели, решающей «Задачу В»***

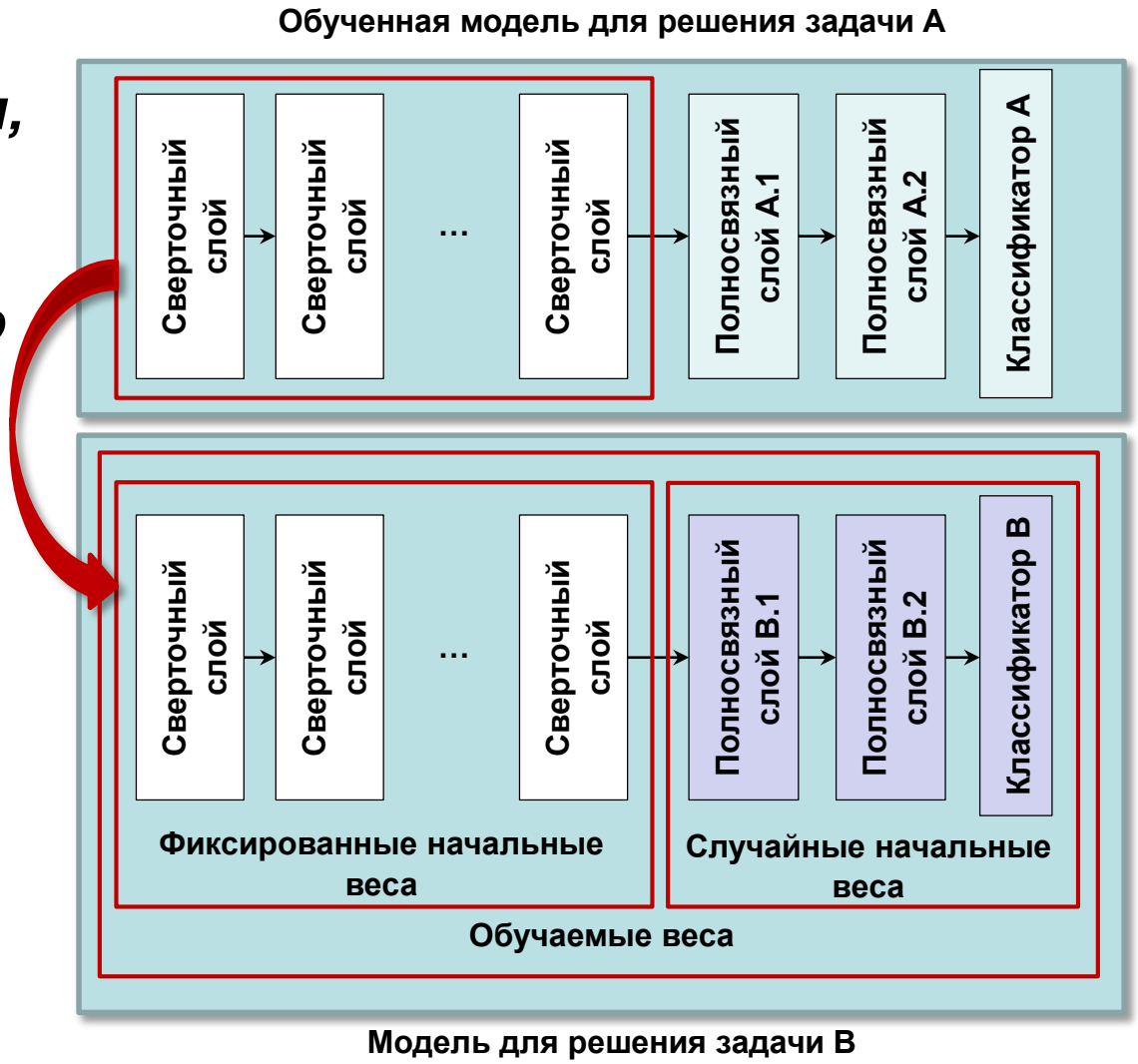


Виды экспериментов (4.1)

- ***Тонкая настройка параметров модели, построенной для решения исходной «Задачи А», с целью решения «Задачи В»***
  - Последние слои глубокой модели, соответствующие классификатору, который решает «Задачу А», заменяются новым классификатором (например, набором полностью связанных слоев с другим количеством выходов)
  - Полученная модель обучается как единая система
  - Эксперимент реализует перенос обучения на основе экземпляров

Виды экспериментов (4.2)

- ***Тонкая настройка параметров модели, построенной для решения исходной «Задачи А», с целью решения «Задачи В»***



Когда/какой тип переноса обучения использовать?

- Основные факторы:
  - Размер тренировочного набора данных целевой задачи
  - Степень сходства данных для обучения моделей целевой и исходной задач

Рекомендации по использованию разных типов переноса обучения (1)

- ***Набор данных целевой задачи небольшой, и он похож на набор данных исходной задачи***

Рекомендации по использованию разных типов переноса обучения (1)

- ***Набор данных целевой задачи небольшой, и он похож на набор данных исходной задачи***
  - Поскольку набор данных небольшой, то не имеет смысла выполнять на нем тонкую настройку весов глубокой модели вследствие возможности переобучения
  - Наиболее правильное решение – использовать начальную часть исходной сети для извлечения признаков и на этих признаках обучать простейший классификатор

Рекомендации по использованию разных типов переноса обучения (2)

- ***Набор данных целевой задачи большой, и он похож на набор данных исходной задачи***

Рекомендации по использованию разных типов переноса обучения (2)

- ***Набор данных целевой задачи большой, и он похож на набор данных исходной задачи***
  - В данном случае можно выполнить тонкую настройку весов сети, ожидая при этом повышение эффективности работы сети на целевой задаче

Рекомендации по использованию разных типов переноса обучения (3)

- ***Набор данных целевой задачи небольшой, и он отличается от набора данных исходной задачи***

Рекомендации по использованию разных типов переноса обучения (3)

- ***Набор данных целевой задачи небольшой, и он отличается от набора данных исходной задачи***
  - Набор данных целевой задачи небольшой, поэтому следует обучать простейший классификатор на признаках, построенных с использованием наиболее ранних слоев исходной сети
  - Ранние слои сети содержат больше преобразований, описывающих особенности набора данных
  - При этом следует выполнять полное обучение сети, т.к. тренировочные наборы данных принципиально отличаются друг от друга

Рекомендации по использованию разных типов переноса обучения (4)

- ***Набор данных целевой задачи большой, и он принципиально отличается от набора данных исходной задачи***

Рекомендации по использованию разных типов переноса обучения (4)

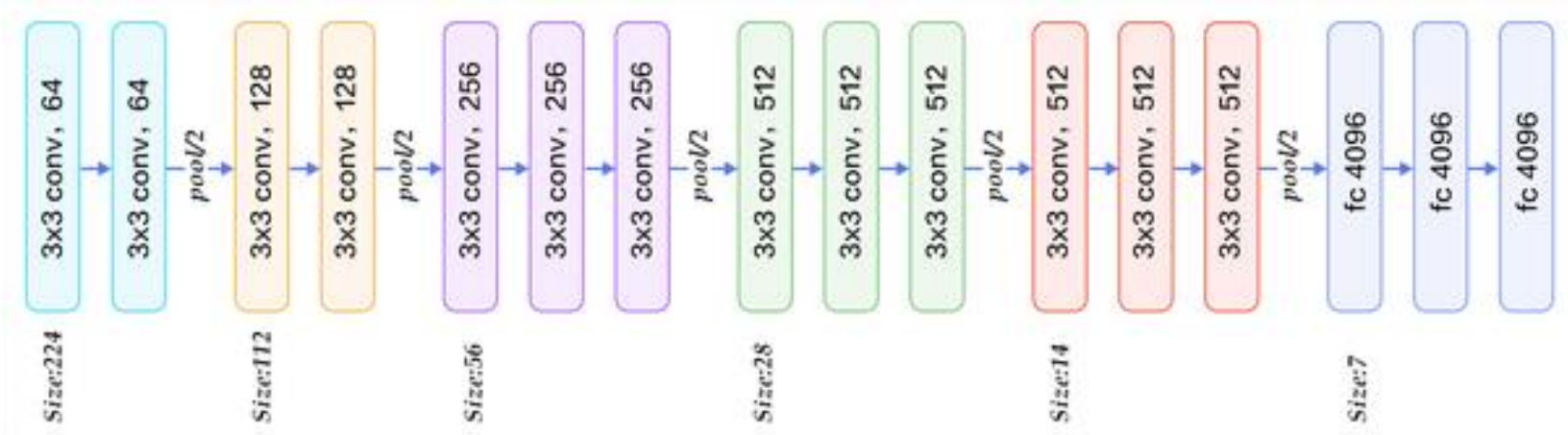
- ***Набор данных целевой задачи большой, и он принципиально отличается от набора данных исходной задачи***
  - В данном случае набор тренировочных данных целевой задачи достаточно большой, чтобы обучить модель, структура которой аналогична модели исходной задачи
  - На практике часто оказывается, что начальная инициализация весов значениями, полученными в ходе решения исходной задачи, позволяет более точно выполнить настройку параметров сети

Пример

ПЕРЕНОС ОБУЧЕНИЯ В INTEL® NEON™ FRAMEWORK

Архитектура сети, используемой для реализации переноса обучения

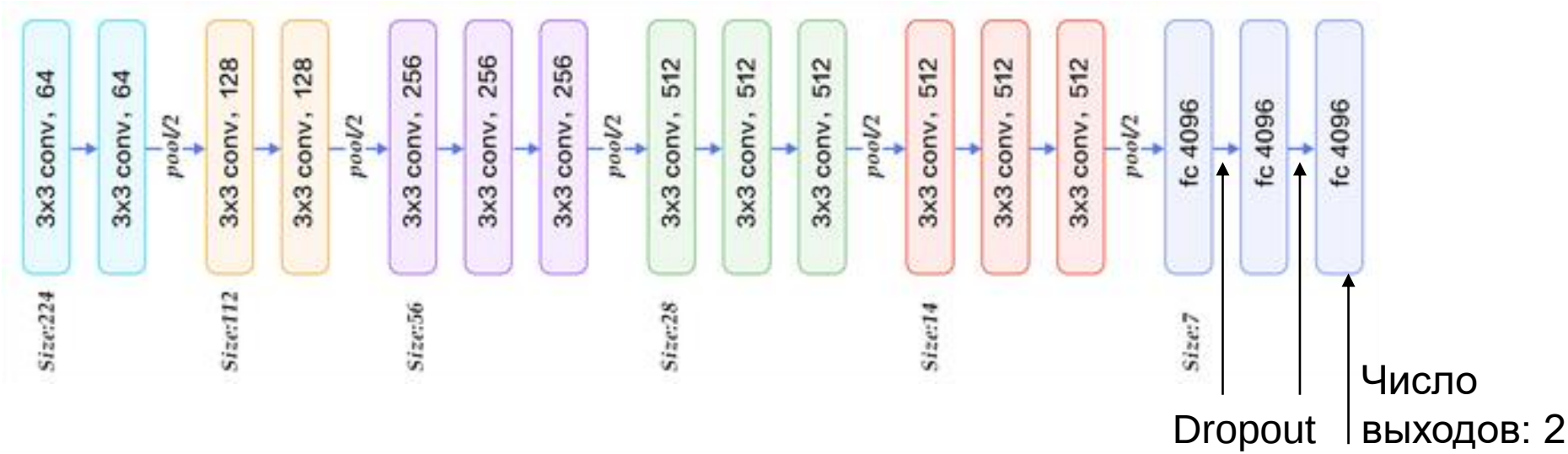
- VGG-16



* Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. – 2015. – [https://arxiv.org/pdf/1409.1556.pdf].

** What is the VGG neural network? [https://www.quora.com/What-is-the-VGG-neural-network].

Модификация архитектуры сети, используемой для реализации переноса обучения



* Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. – 2015. – [https://arxiv.org/pdf/1409.1556.pdf].

** What is the VGG neural network? [https://www.quora.com/What-is-the-VGG-neural-network].

Перенос обучения в Intel® neon™ Framework (1)

```
def generate_vgg_16_transfer_model():
    init1 = Xavier(local=True)
    initfc = GlorotUniform()
    relu = Rectlin()
    conv_params = {'init': init1,'strides': 1,'padding': 1}
    # Set up the model layers
    layers = []

    # set up 3x3 conv stacks with different map sizes
    for i, nofm in enumerate([64, 128, 256, 512, 512]):
        i = i + 1
        layers.append(Conv((3, 3, nofm), **conv_params,
                           name='conv%d_1' % i))
```

Перенос обучения в Intel® neon™ Framework (2)

```
layers.append(Bias(init=Constant(0),
                   name='conv%d_1_bias' % i))
layers.append(Activation(Rectlin()))
layers.append(Conv((3, 3, nofm), **conv_params,
                   name='conv%d_2' % i))
layers.append(Bias(init=Constant(0),
                   name='conv%d_2_bias' % i))
layers.append(Activation(Rectlin()))
if nofm > 128:
    layers.append(Conv((3, 3, nofm), **conv_params,
                       name='conv%d_3' % i))
    layers.append(Bias(init=Constant(0),
                       name='conv%d_3_bias' % i))
```

Перенос обучения в Intel® neon™ Framework (3)

```
        layers.append(Activation(Rectlin()))
        layers.append(Pooling(2, strides=2))

    layers.append(Affine(nout=4096, init=initfc,
                         bias=Constant(0), activation=Rectlin(),
                         name='fc6_new'))
    layers.append(Dropout(keep=0.5))
    layers.append(Affine(nout=4096, init=initfc,
                         bias=Constant(0), activation=Rectlin(),
                         name='fc7_new'))
    layers.append(Dropout(keep=0.5))
    layers.append(Affine(nout=2, init=initfc,
                         bias=Constant(0), activation=Softmax(),
                         name='cls_imdb_wiki_face'))
```

Перенос обучения в Intel® neon™ Framework (4)

```
model = Model(layers=layers)
cost = GeneralizedCost(costfunc=CrossEntropyMulti())
return (model, cost)
```

Перенос обучения в Intel® neon™ Framework (5)

```
if (__name__ == '__main__'):
    args = parse_args(__doc__)
    pretrained_model_filepath = args.source_model
    ...
    target_model, cost = target_model_generator()
    pretrained_model = load_obj(pretrained_model_filepath)
    imported_layers =
      import_matching_layers(pretrained_model, target_model)

    # Make optimizer for transfer learning:
    base_lr = 0.001
    default_optimizer = GradientDescentMomentum(
       base_lr, momentum_coef=0.9,
       stochastic_round=args.rounding, wdecay=0.0005)
```

Перенос обучения в Intel® neon™ Framework (6)

```
optimizers_mapping = {'default': default_optimizer}

# Make small learning rates for imported layers
imported_layers_optimizer_params = \
  default_optimizer.get_description().copy()['config']
imported_layers_optimizer_params['learning_rate'] = 0
imported_layers_optimizer = \
   GradientDescentMomentum(
        **imported_layers_optimizer_params)
optimizers_mapping.update({l: imported_layers_optimizer
   for l in imported_layers})

optimizer = MultiOptimizer(optimizers_mapping)
```

Тестовая инфраструктура

- CPU: Intel® Xeon® CPU E5-2660 0 @ 2.20GHz
- GPU: Tesla K40s 11Gb
- OS: Ubuntu 16.04.4 LTS
- Инструменты:
  - Intel® neon™ Framework 2.6.0
  - CUDA 8.0
  - Python 3.5.2
  - Intel® Math Kernel Library 2017 (Intel® MKL)

Результаты экспериментов

Название	Эксперимент	Параметры обучения	Точность, %	Время обучения, с
TL-1	Обучение весов последних слоев	num_epochs = 90, base_lr = 0.001, lr = 0	85.6	119975
TL-2	Тонкая настройка всех весов	num_epochs = 90, base_lr = 0.001	85.3	119989
TL-3	Полное обучение сети от случайных весов	num_epochs = 30, base_lr = 0.001	**86.3**	39282

Сводные результаты экспериментов

Название	Точность, %	Время обучения, с
FCNN-1	71.2	932
FCNN-2	73.5	977
FCNN-3	**77.7**	1013
CNN-1	79.3	1582
CNN-2	**83.5**	2030
ResNet-18 (90 эпох)	**81.3**	15127
ResNet-50 (30 эпох)	80.9	11849
TL-1	85.6	119975
TL-2	85.3	119989
TL-3	**86.3**	39282

Заключение

- Перенос обучения позволяет быстро получить начальное приближение качества решения поставленной задачи
- Для многих задач использование структуры глубокой модели демонстрирует лучшие показатели качества, чем тонкая настройка параметров сети
- Как правило, время обучения при реализации переноса структуры глубокой модели меньше, чем при тонкой настройке

Основная литература

- Хайкин С. Нейронные сети. Полный курс. – М.: Издательский дом «Вильямс». – 2006. – 1104 с.
- Осовский С. Нейронные сети для обработки информации. – М.: Финансы и статистика. – 2002. – 344 с.
- Goodfellow I., Bengio Y., Courville A. Deep Learning. – MIT Press. – 2016. – [http://www.deeplearningbook.org].

Авторский коллектив

- **Кустикова Валентина Дмитриевна**
  к.т.н., ст.преп. каф. МОСТ ИИТММ,
  ННГУ им. Н.И. Лобачевского
  valentina.kustikova@itmm.unn.ru
- **Жильцов Максим Сергеевич**
  магистрант каф. МОСТ ИИТММ,
  ННГУ им. Н.И. Лобачевского
  zhiltsov.max35@gmail.com
- **Золотых Николай Юрьевич**
  д.ф.-м.н., проф. каф. АГДМ ИИТММ,
  ННГУ им. Н.И. Лобачевского
  nikolai.zolotykh@itmm.unn.ru