Министерство образования и науки Российской Федерации
Московский физико-технический институт (государственный университет)

Физтех-школа прикладной математики и информатики
Кафедра распознавания изображений и обработки текста (РИОТ)

Выпускная квалификационная работа бакалавра

Аугментации в задаче распознавания рукописного текста

Автор:
Студент 020 группы
Курочкин Павел Сергеевич

Научный руководитель:
Упшинский Андрей Леонидович

Москва 2024

Аннотация

Аугментации в задаче распознавания рукописного текста

Курочкин Павел Сергеевич

В современных методах распознавания рукописного текста ключевым элементом является наличие обширного массива аннотированных данных. Однако, в случае ограниченности данных, эффективное применение методов аугментации становится критически важным для повышения точности моделей. Manifold Mixup [1] - современный метод аугментации данных, он представляет собой подход, в котором объединяются два или более изображения или их признаковые карты для создания новых образцов данных. В данном исследовании предлагается метод адаптации Manifold Mixup для использования с подходами, основанными на Connectionist Temporal Classification [2], в контексте задачи распознавания рукописного текста. Проводится всестороннее исследование данного метода, анализируется его влияние на процесс обучения нейронной сети. Результаты исследования показывают значительное улучшение результатов распознавания текста при использовании Manifold Mixup.

Abstract

Augmentations in handwritten text recognition

Содержание

1 Введение **4**
1.1 Проблема 4
1.2 Постановка задачи 5

2 Архитектура решения **6**
2.1 Извлечение визуальных признаков с помощью сверточной сети 6
2.1.1 Residual connection 7
2.1.2 ResNet 8
2.2 Последовательный энкодер 8
2.2.1 Рекуррентные нейронные сети 8
2.2.2 Self-Attention 9
2.3 Декодер 10
2.3.1 Transformer 10
2.3.2 Connectionist Temporal Classification 10

3 Обзор существующих методов аугментации изображений для задачи распознавания рукописного текста **13**

4 Эксперименты **14**

5 Результаты экспериментов **15**

6 Вывод **16**

1 Введение

Распознавание текста является важным этапом в большинстве приложений по анализу изображений документов. Оно позволяет автоматически получать доступ к информации, содержащейся на страницах. За последнее десятилетие произошло огромное улучшение систем распознавания рукописного текста, связанное с появлением новых подходов и архитектур [2], [3], [4], [5], [6], [7], [8].

1.1 Проблема

Использование современных методов нейронных сетей помогает решить проблему высокой стилистической гетерогенности рукописного текста. Однако для таких мощных моделей с большим количеством параметров требуется значительное количество аннотированных изображений для обучения. Было предложено несколько методов, позволяющих уменьшить необходимость аннотированных данных при обучении систем распознавания текста.

Во-первых, обучающий набор можно расширить, добавив синтетические изображения. Это можно сделать, используя рукописные шрифты [9] или создавая изображения текстовых строк на основе индивидуально извлеченных изображений реальных букв [10]. Кроме того, для создания рукописных текстов можно применять генеративно-состязательные сети [11].

Также, при ограниченном объеме данных, необходимо бороться с проблемой переобучения. Для этого было предложено множество подходов к регуляризации, таких как weight decay, dropout [12], batch normalization [21] или раннее прекращение обучения, которые могут быть использованы в процессе обучения сети.

Наконец, для увеличения разнообразия обучающего набора, улучшения обобщающей способности модели и снижения риска переобучения широко применяются методы аугментации данных. Изображения могут быть отражены горизонтально и вертикально, подвергнуты поворотам и угловым трансформациям, изменены по размеру и масштабу. Также возможно добавление шума, обрезка изображений, цветовые трансформации и искажения в различных формах [13], [5]. Подробнее методы аугментации для задачи распознавания рукописного текста описаны в Глава 3.

В [14] для создания новых обьектов было предложено интерполировать признаки обьектов одного класса. В [15] было предложено использовать данный подход для признаков обьектов на выходе промежуточного слоя нейронной сети. В [1] объединяются обе эти идеи и предлагается смешивать целевую переменную и изображения из разных классов, или их промежуточные признаки на различных уровнях сети. Последний подход показывают значительное улучшение качества на новых и состязательных данных. Последний подход был назван Manifold Mixup. В [16] представлен способ адаптации данного подхода для работы с решениями, основанными на Connectionist Temporal Classification [2].

Цель данного исследования заключается в анализе методов аугментации изображений в контексте распознавания рукописного текста и более детальном рассмотрении различных аспектов метода Manifold Mixup. Выбор использования Manifold Mixup обоснован следующими уникальными характеристиками данного метода:

1. Его легкость в обобщении на широкий спектр задач компьютерного зрения, решаемых с помощью нейронных сетей.

2. Согласно результатам исследований [1], [16], данный метод значительно способствует качеству и обобщенности обученной модели.

1.2 Постановка задачи

Рассматриваемые методы аугментации изображений, включая метод Manifold Mixup [1], исследуемый в данной работе, предназначены для преодоления следующих проблем, возникающих в задаче распознавания рукописного текста:

1. Разнообразие изображений рукописного текста из-за различий в стилях авторов и фонах документов.
2. Недостаток аннотированных данных, что затрудняет обобщение моделей.
3. Риск переобучения из-за упомянутых выше проблем и большого количества параметров в сети.

В данной статье проводится анализ различных методов агументаций, решающих данные проблемы. А также исследуются следующие аспекты метода Manifold Mixup:

1. Методы формирования батчей из изображений различных длин и их влияние на стабильность процесса обучения.
2. Роль выбора промежуточного слоя в процессе обучения для различных архитектур.
3. Влияние размера батча на обучение.
4. Воздействие Manifold Mixup на процесс обучения в зависимости от объема имеющихся данных.

В Главе 2 приводится обзор существующих архитектур для решения задачи распознавания рукописного текста. Также представлена архитектура, применяемая в данном исследовании, обосновывается выбор данной архитектуры.

Глава 3 посвящена обзору существующих методов аугментации изображений для задачи распознавания рукописного текста, включая Manifold Mixup. Оценивается эффективность их применения для решения вышеупомянутых задач.

В Главе 4 представлено описание практической части и проведенных экспериментов.

Глава 5 анализирует различные аспекты метода Manifold Mixup на основе проведенных экспериментов

И, наконец, в Главе 6 содержатся выводы из проведенного исследования.

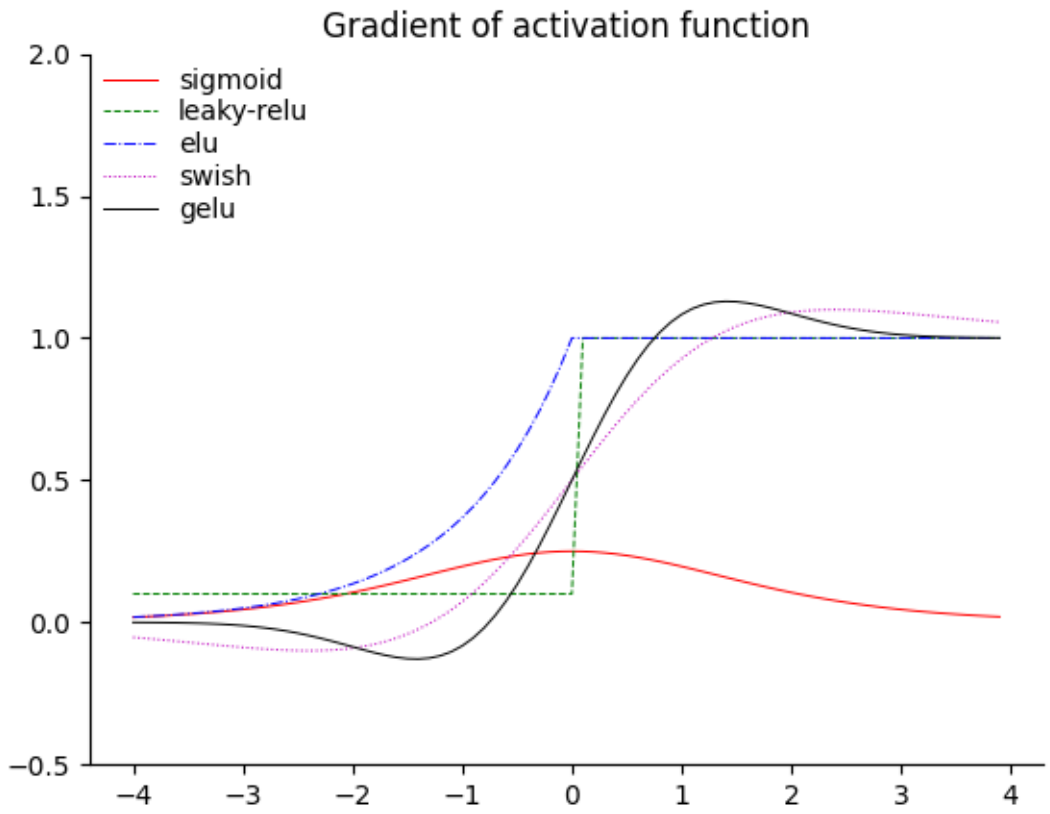

Рис. 1: Градиенты некоторых популярных функций активации.

2 Архитектура решения

Современные модели, применяемые в задаче распознавания рукописного текста, обычно включают в себя три ключевых компонента:

1. Сверточную нейронную сеть, используемую для извлечения признаков из входных данных.

2. Последовательный энкодер

3. Декодер, который завершает процесс, трансформируя закодированные данные в окончательную текстовую транскрипцию с учетом информации, полученной от энкодера.

2.1 Извлечение визуальных признаков с помощью сверточной сети

Подобно другим задачам компьютерного зрения, сверточная сеть используется для извлечения соответствующих визуальных признаков из текстовых изображений. Следующие архитектуры являются популярным выбором: ResNet [17], Inception [18], MobileNet [19]. Увеличение сложности сверточной сети, как правило, приводит к умеренному приросту точности модели [20]. В этой работе используется архитектура ResNet.

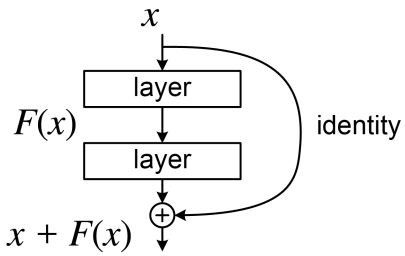

Рис. 2: Residual блок.

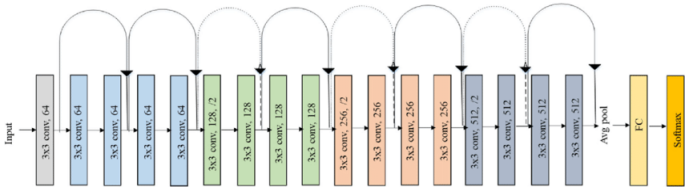

Рис. 3: Архитектура ResNet-18.

2.1.1 Residual connection

При обучении глубоких моделей градиент имеет тенденцию становиться очень маленьким: это называется проблема исчезающего градиента. Это связано с тем, что градиент проходит через ряд слоев, каждый из которых может уменьшить его. Например, градиент многих популярных функций активации приближается к нулю на значительной части числовой прямой [Рис 1].

Одним из решений проблемы исчезающего градиента является использование в качестве слоев нейронной сети residual блока [Рис 2], определенного следующим образом:

$$\mathcal{F}_l^{'}(x) = \mathcal{F}_l(x) + x \tag{1}$$

где $\mathcal{F}_l$ - стандартное нелинейное отображение (например, линейное - функция активации - линейное). Зачастую легче научиться предсказывать небольшое возмущение на входе, чем результат напрямую.

Модель с residual блоком имеет такое же количество параметров, как и модель без него, но ее легче тренировать. Причина в том, что градиенты могут перетекать непосредственно из выходных данных в более ранние слои. Чтобы убедиться в этом, рассмотрим градиент функции ошибки по параметрам слоя l. Имеем

$$z_L = z_l + \sum_{i=l}^{L-1} \mathcal{F}_i(z_i; \theta_i) \tag{2}$$

где z_i - признаки на выходе из i-го слоя сети. Таким образом, мы можем вычислить градиент функции потерь относительно параметров l-го слоя следующим образом:

$$\frac{\partial \mathcal{L}}{\partial \theta_l} = \frac{\partial z_l}{\partial \theta_l} \frac{\partial \mathcal{L}}{\partial z_L} (1 + \sum_{i=l}^{L-1} \frac{\partial \mathcal{F}_i(z_i; \theta_i)}{\partial z_l}) \tag{3}$$

Таким образом, мы видим, что градиент на слое l напрямую зависит от градиента на слое L, причем независимо от глубины сети.

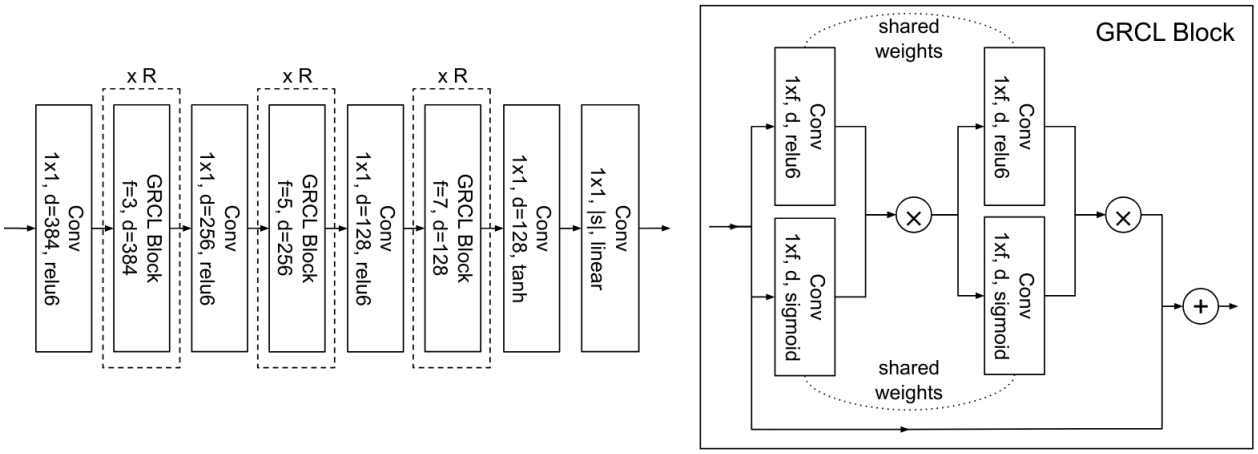

Рис. 4: Архитектура GRCL.

2.1.2 ResNet

Победителем конкурса по компьютерному зрению ImageNet 2015 года стала команда Microsoft, предложившая модель, известную сейчас как ResNet [Рис 3]. Данная модель состоит из residual блоков, имеющих следующий вид: свертка-BN-relu-свертка-BN, где BN - батч нормализация [21]). Подобная архитектура позволяет обучать очень глубокие модели, а также решает проблему затухания градиентов. Именно из-за указанных преимуществ, а также из практического опыта, данная архитектура была отобрана для проведения данного исследования.

2.2 Последовательный энкодер

Сверточная сеть, предназначенная для извлечения визуальных признаков, имеет ограниченное рецептивное поле, что ограничивает ее способность учитывать широкий контекст информации. Это создает сложности при обработке длинных последовательностей в сложных сценариях, таких как распознавание рукописного текста. Для учета контекста на больших расстояниях используется энкодер. Существует много различных архитектур энкодера, далее будут рассмотрены основные из них.

2.2.1 Рекуррентные нейронные сети

Рекуррентная нейронная сеть — это нейронная сеть, которая отображает входное пространство последовательности в выходное пространство последовательностей с сохранением состояния. То есть элемент выходной последовательности y_t зависит не только от элемента входной последовательности x_t, но и от скрытого состояния системы h_t, которое обновляется. Для простоты обозначений пусть T будет длиной вывода (с учетом того, что она выбирается динамически). Тогда рекурентная сеть соответствует следующей условной генеративной модели:

$$p(y_{1:T}|x) = \sum_{h_{1:T}} p(y_{1:T}, h_{1:T}|x) = \sum_{h_{1:T}} \prod_{t=1}^{T} p(y_t|h_t)p(h_t|h_{t-1}, y_{t-1}, x) \tag{4}$$

где h_t — скрытое состояние, и где мы определяем $p(h_1|h_0, y_0, x) = p(h_1|x)$ как начальное скрытое состояние. Мы предполагаем, что скрытое состояние вычисляется детерминированно следующим образом:

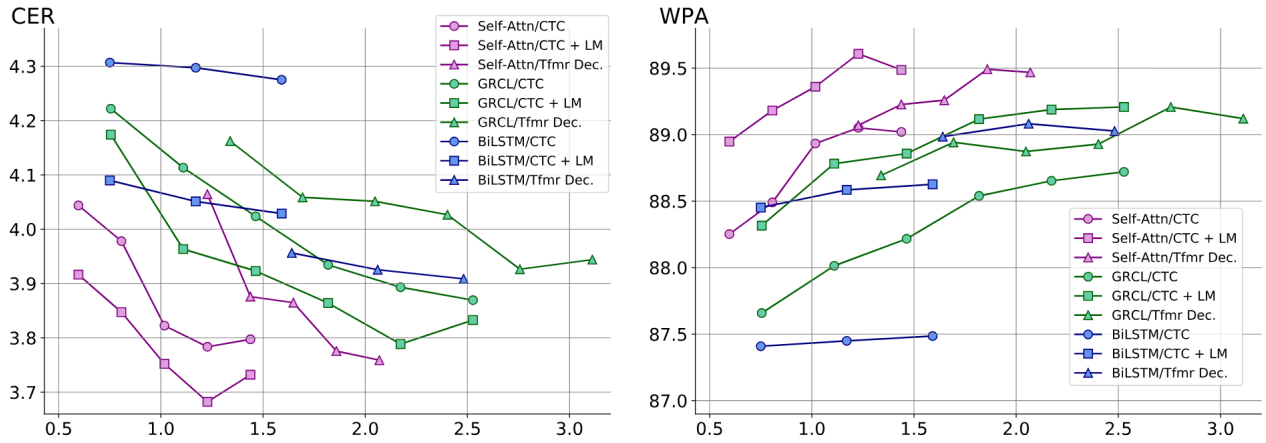

Рис. 5: Сравнение различных архитектур для задачи распознавания текста. Взято из [20].

$$p(h_t|h_{t-1}, y_{t-1}, x) = \mathbb{I}(h_t = f(h_{t-1}, y_{t-1}, x)) \tag{5}$$

для некоторой детерминированной функции f. Функция обновления f обычно задается выражением

$$h_t = \varphi(W_{xh}[x; y_{t-1}] + W_{hh}h_{t-1} + b_h) \tag{6}$$

Рекурентные сети с достаточным количеством скрытых единиц в принципе могут запоминать входные данные из далекого прошлого. Однако сети с "ванильной"архитектурой не могут этого сделать из-за проблемы исчезающего градиента. Существует архитектурное решение данной проблемы, в котором мы обновляем скрытое состояние аддитивным способом, аналогично residual блокам в ResNet: GRU и LSTM.

Различные варианты рекуррентных сетей использовались для решения задачи распознавания рукпоисного текста: LSTM [5] [4], BiLSTM [6] [7], Gated Recurrent Convolution Layer (GRCL) [Рис 4] [22].

2.2.2 Self-Attention

Self-Attention энкодер, предложенный в [3] широко используется в задачах из NLP и компьютерного зрения. Распознавание текстовых строк, как задача преобразования изображения в последовательность, не является исключением. Self-attention способен улавливать долгосрочные зависимости в последовательностях лучше, чем рекуррентные сети, благодаря возможности обрабатывать взаимодействия между всеми элементами последовательности одновременно. Также, в отличие от рекуррентных сетей, self-attention не сталкивается с проблемой затухания градиентов.

Выход сверточной сети, с удаленным измерением высоты изображения ($X \in \mathbb{R}^{n \times d}$), поступает в энкодер. Выход Y слоя Attention получается следующим образом:

$$\begin{aligned} Q &= XW_q \\ K &= XW_k \\ V &= XW_v \\ Y &= softmax(\frac{QK^T}{\sqrt{d}}V) \end{aligned} \tag{7}$$

где W_q, W_k и W_v — матрицы параметров размера $d \times d$, которые задают проекцию входной последовательности X в пространство запросов, ключей и значений соответственно. Закодированный признак Y представляет собой выпуклую комбинацию вычисленных значений V, матрица сходства вычисляется как скалярное произведение запросов и ключей.

Эффективность "ванильного"Attention может быть низкой, поскольку данный слой инвариантен к перестановкам, и, следовательно, игнорирует порядок элементов входной последовательности. Чтобы преодолеть эту проблему, к признакам элементов входной последовательности добавлятся информация о позиции элемента (positional embedding). Можно представить positional embedding как матрицу $\mathbf{P}^{n \times d}$.

В [3] было предложено использовать синусоидальный базис следующего вида:

$$p_{i,2j} = \sin(\frac{i}{C^{\frac{2j}{d}}}), \quad p_{i,2j+1} = \cos(\frac{i}{C^{\frac{2j}{d}}}) \tag{8}$$

где C соответствует некоторой максимальной длине последовательности. Одним из важных плюсов данного представления является то, что представление одной позиции линейно предсказуемо относительно любой другой, если известно их относительное расстояние. В частности, выполняется $p_{t+\phi} = f(p_t)$, где f - некоторое линейное отображение. А именно

$$\begin{pmatrix} \sin(\omega_k(t+\phi)) \\ \cos(\omega_k(t+\phi)) \end{pmatrix} = \begin{pmatrix} \cos(\omega_k\phi) & \sin(\omega_k\phi) \\ -\sin(\omega_k\phi) & \cos(\omega_k\phi) \end{pmatrix} \begin{pmatrix} \sin(\omega_k t) \\ \cos(\omega_k t) \end{pmatrix} \tag{9}$$

То есть при маленьком ϕ имеем $p_{t+\phi} \approx p_t$. Positional embedding, как правило, прибавляется ко входу, то есть: $PosEmbed(X) = X + \mathbf{P}$.

Существуют также другие варианты. Например, relative positional embedding [23].

2.3 Декодер

Декодеры берут признаки закодированной последовательности и пытаются декодировать ее текстовое содержимое. Архитектуры декодеров для рекурентных сетей были описаны ранее. Два других популярных подхода к решению этой задачи: transformer [3] и Connectionist Temporal Classification [2] декодеры.

2.3.1 Transformer

Декодер Transformer стал предпочтительным декодером для задач прогнозирования последовательности, таких как машинный перевод. Он также широко применяется в задаче распознавания рукописного текста. Например, state-of-the-art решение согласно бенчмарку IAM принадлежит на момент написания этого текста decoder-only transformer архитектуре [8]. Минус данной архитектуры - это то, что она является более громоздкой. Она требует большего количества параметров и, соотвественно, данных для обучения [Рис 5].

2.3.2 Connectionist Temporal Classification

Декодер Connectionist Temporal Classification изначально использовался для распознавания речи, позже исследователи добились огромного успеха, применив его для задач распознавания текста. Далее представлен математический формализм временной классификации, а также ошибки, используемой в качестве метрики в данном исследовании.

	Rect.	Encoder / Decoder	Train Dataset	IAM ↓	RIMES ↓	IIIT5K ↑	SVT ↑	IC13 ↑	IC15 ↑
Bluche and Messina [6]	No	GCRNN/CTC	IAM (50k lexicon)	3.2	**1.9**	-	-	-	-
Michael et al. [37]	No	LSTM/LSTM w/Attn	IAM	4.87	-	-	-	-	-
Kang et al. [28]	No	Transformer	IAM	4.67	-	-	-	-	-
Bleeker and de Rijke [4]	No	Transformer	MJ + ST	-	-	94.7	89.0	93.4	75.7
Lu et al. [34]	No	Global Context Attn / Tfmr Dec.	MJ + ST + SA	-	-	95.0	90.6	95.3	79.4
Qiao et al. [42]	No	LSTM / LSTM + Attn	MJ + ST	-	-	94.4	90.1	93.3	77.1
Sheng et al. [45]	Yes	S-Attn / Attn	MJ + ST	-	-	93.4	89.5	91.8	76.1
Shi et al. [48]	Yes	LSTM / LSTM + Attn	MJ + ST	-	-	91.93	93.49	89.75	-
Wang et al. [55]	No	FCN / GRU	MJ + ST	6.4	2.7	94.3	89.2	93.9	74.5
SCATTER [32]	Yes	CNN / BiLSTM	MJ + ST + SA	-	-	93.9	92.7	94.7	**82.8**
Yu et al. [57]	No	Attn / Semantic Attn.	MJ + ST	-	-	94.8	91.5	95.5	82.7
Ours	No	S-Attn / CTC	MJ + ST + Public	-	-	92.06	87.94	92.15	72.36
	No	S-Attn / CTC + LM	MJ + ST + Public	-	-	93.63	91.50	93.61	74.77
	No	Transformer	MJ + ST + Public	-	-	93.93	92.27	93.88	77.08
	No	S-Attn / CTC	Internal	4.62	10.80	96.26	91.96	94.43	78.43
	No	S-Attn / CTC + LM	Internal	3.15	7.79	**96.83**	**94.59**	**95.98**	80.36
	No	Transformer	Internal	3.99	9.71	96.54	92.59	94.34	79.68
	No	S-Attn / CTC	Internal + Public	3.53	2.48	95.66	91.34	93.70	78.38
	No	S-Attn / CTC + LM	Internal + Public	**2.75**	1.99	96.43	93.66	95.25	79.92
	No	Transformer	Internal + Public	2.96	2.01	96.44	92.50	93.97	80.45

Рис. 6: Сравнение различных архитектур для задачи распознавания текста на различных бенмарках.
Взято из [20].

Пусть S - это набор обучающих примеров, выбранных из фиксированного распределения $D_{\mathcal{X}\times\mathcal{Z}}$. Пространство входных данных $\mathcal{X} = (\mathbb{R}^m)^*$ представляет собой множество всех последовательностей из m-мерных векторов вещественных чисел. Целевое пространство $\mathcal{Z} = L^*$ представляет собой множество всех последовательностей из (конечного) алфавита L. В общем случае, мы обозначаем элементы L^* как последовательности символов. Каждый пример в S состоит из пары последовательностей (x, z). Целевая последовательность $z = (z_1, z_2, ..., z_U)$ не длиннее входной последовательности $x = (x_1, x_2, ..., x_T)$, то есть $U \leq T$. Поскольку входные и целевые последовательности обычно не имеют одинаковой длины, нет априорного способа их выравнивания.

Цель состоит в использовании S для обучения временного классификатора $h : \mathcal{X} \to \mathcal{Z}$, который классифицирует ранее не виденные входные последовательности таким образом, чтобы минимизировать некоторую специфическую ошибку задачи.

Общепризнанным выбором является следующая ошибка: для заданного тестового набора $S_0 \subset D_{\mathcal{X}\times\mathcal{Z}}$, не пересекающегося с S, определяется коэффициент ошибок меток (label error rate, LER) временного классификатора h как нормализованное расстояние Левенштейна между его предсказаниями и ответом на S_0, т.е.:

$$LER(h, S_0) = \frac{1}{Z} \sum_{(x,z)\in S_0} ED(h(x), z) \tag{10}$$

где Z - общее количество целевых меток в S_0, а $ED(p, q)$ - расстояние Левенштейна между двумя последовательностями p и q, т.е. минимальное количество вставок, замен и удалений, необходимых для преобразования p в q. Это естественная метрика для задач (таких как распознавание речи или рукописного текста), где целью является минимизация частоты ошибок транскрипции. Популярными метриками в задаче распознавания рукописного текста являются WER (word error rate) и CER (character error rate) - описанное выше расстояние Левенштейна на уровне слов и символов соответственно.

Также в данном исследовании будет использоваться метрика LabelAccuracy (Label

может быть словом, строкой или фрагментом). Определяется она так:

$$LabelAcc(h, S_0) = 100 * (1 - \frac{\sum_{(x,z)\in S_0} ED(h(x), z)}{\sum_{(x,z)\in S_0} |z|}) \tag{11}$$

Пусть мы имеем некоторую нейронную сеть $N_w : (\mathbb{R}^m)^T \to (\mathbb{R}^n)^T$. Пусть $y = N_w(x)$ - последовательность выходов сети, и обозначим y_t^k активацию выходного узла k в момент времени t (последний слой softmax). Тогда y_t^k интерпретируется как вероятность наблюдения символа k в момент времени t, что определяет распределение над множеством L_0^T длины T последовательностей алфавита $L_0 = L \cup \{blank\}$. Тут мы добавляем в алфавит символ $blank$, обозначающий пропуск. Элементы L_0^T обычно называют путями.

$$p(\pi|x) = \prod_{t=1}^{T} y_t^{\pi_t}, \quad \forall \pi \in L_0^T \tag{12}$$

Далее определим отображение $B : L_0^T \to L_{\leq T}$, где $L_{\leq T}$ - это множество последовательностей длиной менее или равной T по оригинальному алфавиту L. Мы делаем это, просто удаляя все пробелы и повторяющиеся символы из путей. Интуитивно это соответствует выводу нового символа, когда сеть переходит от предсказания отсутствия символа к предсказанию символа, или от предсказания одного символа к другому. Наконец, используем B для определения условной вероятности данного предсказания $I \in L_{\leq T}$ как сумму вероятностей всех путей, соответствующих ему:

$$p(I|x) = \sum_{\pi \in B^{-1}(I)} p(\pi|x) \tag{13}$$

Вывод классификатора должен быть наиболее вероятным предсказанием для входной последовательности:

$$h(x) = \arg\max_{I \in L_{\leq T}} p(I|x) \tag{14}$$

В данном исследовании поиск такого предсказания строится на предположении, что наиболее вероятный путь будет соответствовать наиболее вероятному предсказанию:

$$\begin{aligned} h(x) &\approx B(\pi^*) \\ \pi^* &= \arg\max_{\pi \in L_0^T} p(\pi|x). \end{aligned} \tag{15}$$

Это легко найти, так как π^* представляет собой просто конкатенацию самых вероятных символов на каждом этапе. Такой подход не гарантирует нахождение наиболее вероятного предсказания, но достаточно хорошо его приближает.

Модель обучается методом максимального правдоподобия. Для вычисления условных вероятностей $p(I|x)$ отдельных предсказаний используется динамическое программирование [2]. Также включение явной языковой модели поверх логитов может значительно повысить точность [24].

Во время выбора архитектуры учитывался собственный практический опыт, а также результаты исследований по данной теме. Например, в [20] проводится сравнительный анализ различных архитектур энкодеров / декодеров [Рис 6]. Было решено выбрать в качестве:

1. Сверточной нейронной сети: ResNet
2. Последовательного энкодера: TransformerEncoder
3. Декодера: Connectionist Temporal Classification

3 Обзор существующих методов аугментации изображений для задачи распознавания рукописного текста

взять отсюда (3)

4 Эксперименты

Требуется разбить большую задачу, описанную в постановке, на более мелкие подзадачи. Процесс декомпозиции следует продолжать до тех пор, пока подзадачи не станут достаточно простыми для решения непосредственно. Это может быть достигнуто, например, путем проведения эксперимента, доказательства теоремы или поиска готового решения.

5 Результаты экспериментов

Если в рамках работы писался какой-то код, здесь должно быть его описание: выбранный язык и библиотеки и мотивы выбора, архитектура, схема функционирования, теоретическая сложность алгоритма, характеристики функционирования (скорость/-память).

6 Вывод

Здесь надо перечислить все результаты, полученные в ходе работы. Из текста должно быть понятно, в какой мере решена поставленная задача.

Список литературы

[1] Manifold Mixup: Better Representations by Interpolating Hidden States / Verma Vikas, Lamb Alex, Beckham Christopher et al. // *arXiv:1806.05236*. — 2018.

[2] Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks / Graves Alex, Fern Santiago, Gomez Faustino, Schmidhuber J¨urgen // International Conference on Machine Learning. — 2006.

[3] Attention Is All You Need / Vaswani Ashish, Shazeer Noam, Parmar Niki et al. // *arXiv:1706.03762*. — 2017.

[4] *Alex, Graves*. Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks / Graves Alex, Fern, Schmidhuber J¨urgen // Conference on Neural Information Processing Systems. — 2008.

[5] The A2iA Multi-lingual Text Recognition System at the Second Maurdor Evaluation / Bastien Moysset, Théodore Bluche, Maxime Knibbe et al. // 14th International Conference on Frontiers in Handwriting Recognition. — 2014.

[6] *Puigcerver, Joan*. Are Multidimensional Recurrent Layers Really Necessary for Handwritten Text Recognition? / Joan Puigcerver // International Conference on Document Analysis and Recognition. — 2017.

[7] *Bluche, Theodore*. Gated Convolutional Recurrent Neural Networks for Multilingual Handwriting Recognition / Theodore Bluche, Ronaldo Messina // International Conference on Document Analysis and Recognition. — 2017.

[8] *Fujitake, Masato*. DTrOCR: Decoder-only Transformer for Optical Character Recognition / Masato Fujitake // *arXiv:2308.15996*. — 2023.

[9] *Helmers, Muriel*. Generation and Use of Synthetic Training Data in Cursive Handwriting Recognition / Muriel Helmers, Horst Bunke // Iberian Conference on Pattern Recognition and Image Analysis. — 2003.

[10] *Shen, Xi*. A Method of Synthesizing Handwritten Chinese Images for Data Augmentation / Xi Shen, Ronaldo Messina // 15th International Conference on Frontiers in Handwriting Recognition (ICFHR). — 2016.

[11] *Alonso, Eloi*. Adversarial Generation of Handwritten Text Images Conditioned on Sequences / Eloi Alonso, Bastien Moysset, Ronaldo Messina // *arXiv:1903.00277*. — 2019.

[12] Dropout: A Simple Way to Prevent Neural Networks from Overfitting / Sutskever Ilya, Krizhevsky Alex, Hinton Geoffrey et al. // *Journal of Machine Learning Research 15*. — 2014.

[13] Data Augmentation for Recognition of Handwritten Words and Lines Using a CNN-LSTM Network / Curtis Wigington, Seth Stewart, Brian Davis et al. // 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). — 2017.

[14] SMOTE: Synthetic Minority Over-sampling Technique / Chawla V. N., Bowyer W. K., Hall O. L., Kegelmeyer P. W. // *arXiv:1106.1813*. — 2011.

[15] *Terrance, DeVries*. Dataset Augmentation in Feature Space / DeVries Terrance, Taylor W. Graham // *arXiv:1702.05538*. — 2017.

[16] *Moysset, Bastien*. Manifold Mixup improves text recognition with CTC loss / Bastien Moysset, Ronaldo Messina // *arXiv:1903.04246*. — 2019.

[17] Deep Residual Learning for Image Recognition / He Kaiming, Zhang Xiangyu, Ren Shaoqing, Sun Jian // *arXiv:1512.03385*. — 2015.

[18] Going Deeper with Convolutions / Szegedy Christian, Liu Wei, Jia Yangqing et al. // *arXiv:1409.4842*. — 2014.

[19] MobileNetV2: Inverted Residuals and Linear Bottlenecks / Sandler Mark, Howard Andrew, Zhu Menglong et al. // *arXiv:1801.04381*. — 2018.

[20] Rethinking Text Line Recognition Models / Diaz Hernandez Daniel, Qin Siyang, Ingle Reeve et al. // *arXiv:2104.07787*. — 2021.

[21] *Sergey, Ioffe*. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift / Ioffe Sergey, Szegedy Christian // *arXiv:1502.03167*. — 2015.

[22] Gated recurrent convolution neural network for ocr. / In I. Guyon, U. V. Luxburg, S. Bengio et al. // Advances in Neural Information Processing Systems. — Curran Associates, 2017.

[23] *Peter, Shaw*. Self-Attention with Relative Position Representations / Shaw Peter, Uszkoreit Jakob, Vaswani Ashish // *arXiv:1803.02155*. — 2018.

[24] Sequence-to-Label Script Identification for Multilingual OCR / Fujii Yasuhisa, Driesen Karel, Baccash Jonathan et al. // *arXiv:1708.04671*. — 2017.