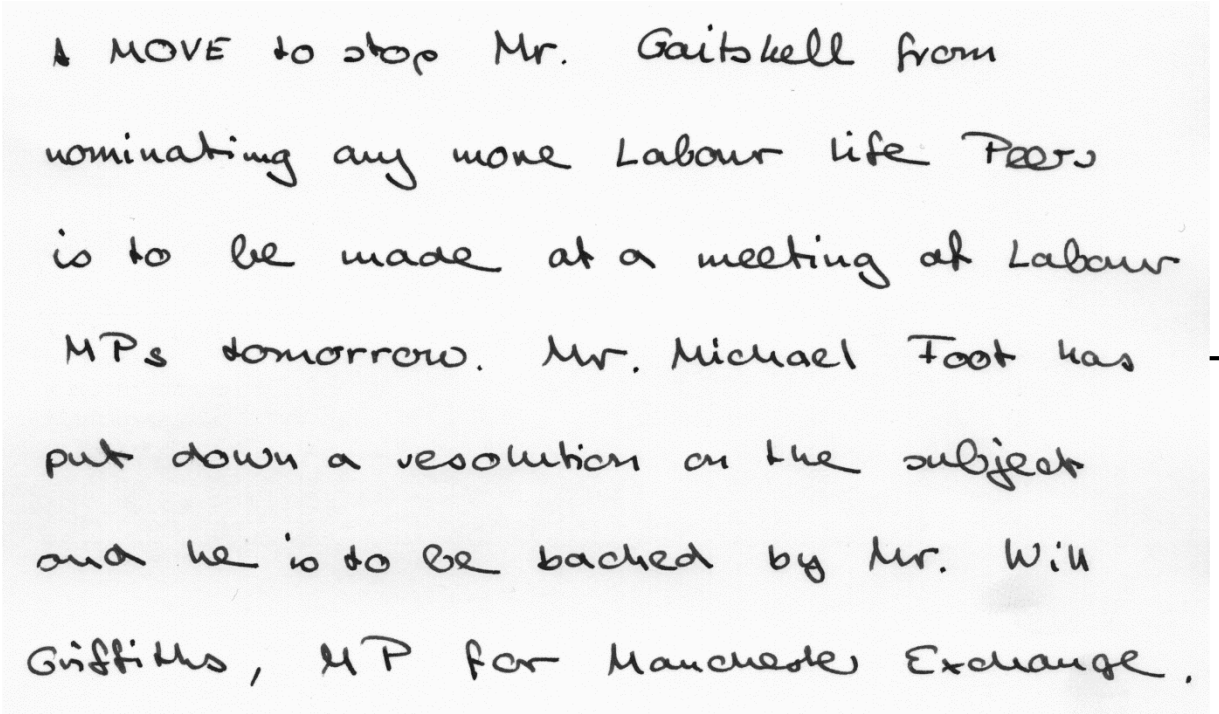
Распознавание рукописного текста

Выполнил студент: Красников Р.А.
Группа: 5040102/40101

Преподаватель: Иванов Д.Ю.

Постановка задачи

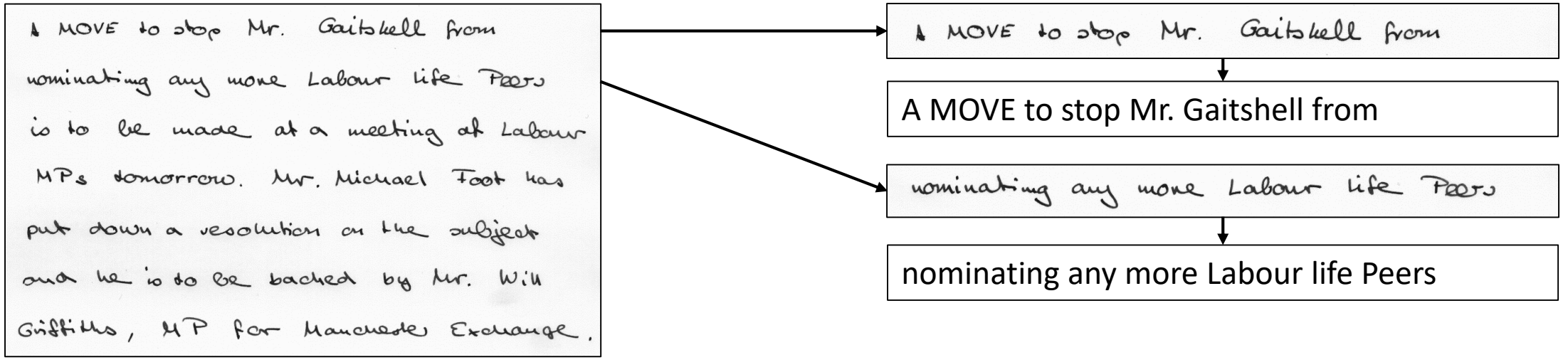
- Задача: распознавание рукописного текста - по изображению документа с рукописным текстом распознать и вывести его содержимое
- Вход: изображение документа
- Выход: содержимое документа (текст)



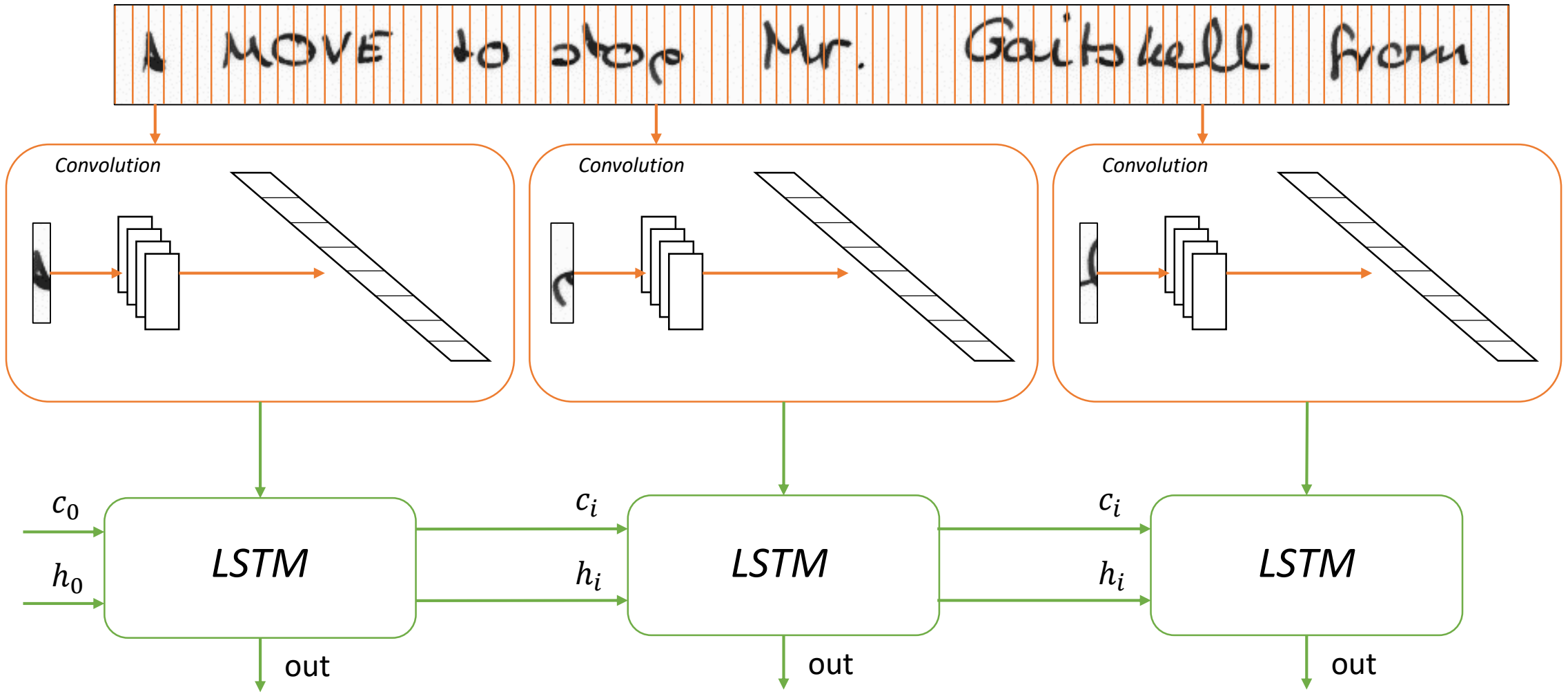
A MOVE to stop Mr. Gaitskell from nominating any more Labour life Peers...

Выбранная модель машинного обучения

- Сверточно-рекурсивная нейронная сеть (Convolutional-Recurrent Neural Network - CRNN)
  - Набор сверточных слоев, переводящих части строки текста в набор карт признаков
  - Последовательность наборов карт признаков обрабатывается LSTM (Long Short Term Memory)
  - Датасет: IAM - набор рукописных текстов на английском языке
  - Пример подобной архитектуры: https://habr.com/ru/articles/720614/
- Фото документа разбивается на строки с помощью инструментов OpenCV без использования нейросетевых подходов
- Каждая строка обрабатывается нейронной сетью
- Результаты склеиваются в итоговый текст с разделителями \n.

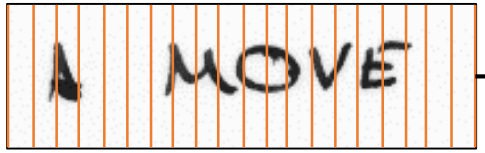


Сверточно-рекурсивная нейронная сеть



Нейронная сеть: формат выхода

- Алфавит: “ABCDE...” и “+” - специальный символ
- Выход модели по длине совпадает с числом вертикальных полос, на которые разбивается строчка
- Символ “+” наделяется семантикой “Читаю очередной символ”
- За счет механизма запоминания удается накапливать информацию о прочитанных частях символа
- Кода символ прочитан, срабатывает механизм забывания, и на выход идет прочитанный символ



→ +|+A+++|+++MM++O++V+++EE++|

- Характерная проблема таких моделей - потенциальное дублирование символов. Оно заключается в том, что один символ встречается на нескольких чанках и из-за отсутствия четких границ между символами в картах признаков выдается моделью на выход несколько раз
- Качественно проблема решается с помощью CTC (Connectionist Temporal Classification) Decoder’a
  - Существуют различные умные нейросетевые подходы, дополнительно исправляющие опечатки и другие возможные ошибки
  - В данном проекте используется обыкновенная жадная декодировка

+|+A+++|+++MM++O++V+++EE++| → A MOVE

Нейронная сеть: используемая архитектура

- Проект реализован с помощью PyTorch
- Приведен вывод torchinfo для батча из 64 изображений
- Функции активации LeakyReLU
- На выходе LogSoftmax (нужно для CTC Loss из torch)

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
IamSentencesCRNN                         [64, 140, 80]             --
├─Sequential: 1-1                        [64, 256, 1, 140]         --
│    └─Conv2d: 2-1                       [64, 32, 32, 560]         832
│    └─LeakyReLU: 2-2                    [64, 32, 32, 560]         --
│    └─MaxPool2d: 2-3                    [64, 32, 16, 280]         --
│    └─Conv2d: 2-4                       [64, 64, 16, 280]         51,264
│    └─LeakyReLU: 2-5                    [64, 64, 16, 280]         --
│    └─MaxPool2d: 2-6                    [64, 64, 8, 280]          --
│    └─Conv2d: 2-7                       [64, 64, 8, 280]          36,928
│    └─LeakyReLU: 2-8                    [64, 64, 8, 280]          --
│    └─MaxPool2d: 2-9                    [64, 64, 4, 280]          --
│    └─BatchNorm2d: 2-10                 [64, 64, 4, 280]          128
│    └─Conv2d: 2-11                      [64, 128, 4, 280]         73,856
│    └─LeakyReLU: 2-12                   [64, 128, 4, 280]         --
│    └─MaxPool2d: 2-13                   [64, 128, 2, 140]         --
│    └─Conv2d: 2-14                      [64, 128, 2, 140]         147,584
│    └─LeakyReLU: 2-15                   [64, 128, 2, 140]         --
│    └─BatchNorm2d: 2-16                 [64, 128, 2, 140]         256
│    └─Conv2d: 2-17                      [64, 256, 2, 140]         295,168
│    └─LeakyReLU: 2-18                   [64, 256, 2, 140]         --
│    └─MaxPool2d: 2-19                   [64, 256, 1, 140]         --
│    └─Conv2d: 2-20                      [64, 256, 1, 140]         590,080
│    └─LeakyReLU: 2-21                   [64, 256, 1, 140]         --
├─LSTM: 1-2                              [64, 140, 256]            790,528
├─Linear: 1-3                            [64, 140, 80]             20,560
├─LogSoftmax: 1-4                        [64, 140, 80]             --
==========================================================================================
Total params: 2,007,184
Trainable params: 2,007,184
Non-trainable params: 0
Total mult-adds (Units.GIGABYTES): 46.55
==========================================================================================
Input size (MB): 4.59
Forward/backward pass size (MB): 739.74
Params size (MB): 8.03
Estimated Total Size (MB): 752.35
==========================================================================================
```

Функция потерь и метрика точности

- Для использованной архитектуры наиболее распространенной функцией потерь является CTC (Connectionist Temporal Classification) Loss
- CTC Loss - функция потерь для задач, где неизвестно точное соответствие между временными шагами входа и символами выхода. CTC вычисляет вероятность целевой последовательности как сумму вероятностей всех возможных временных выравниваний с использованием специального пустого символа (blank).

$$P(\text{“AB”}) = P(\text{“+++AB”}) + P(\text{“+A+B+”}) + \ldots + P(\text{“AB+++”})$$

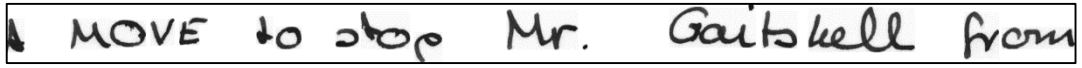
- Метрика точности CER (Character Error Rate): расстояние Левенштейна, нормированное на длину точной метки
- В качестве точности взята величина (1 - CER) * 100 [%]
- В отличие от CTC Loss более осязаема и работает не с логарифмами вероятностей, а с вполне конкретными объектами: выходом модели и меткой, например, “A grt hause” и “A great house”
- Тем не менее, зависит от CTC Decoder’a, поэтому, строго говоря точность самой нейронной сети оценивает несколько опосредованно!

$$\text{CER} = \text{levenstein(out, label)} / \text{len(label)}$$

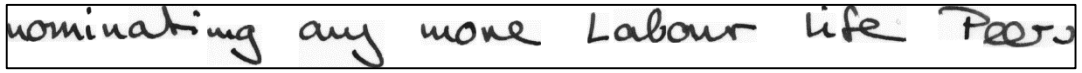
Датасет: iam-sentences

- Датасет iam-sentences: постобработка датасета IAM, в которой рукописные документы поделены на строки и размечены, найден на Kaggle: https://www.kaggle.com/datasets/debadityashome/iamsentences
- Содержит изображения рукописных строк и файл с разметкой

a01-000u-s00-00.png



a01-000u-s00-01.png



a01-000u-s00-02.png

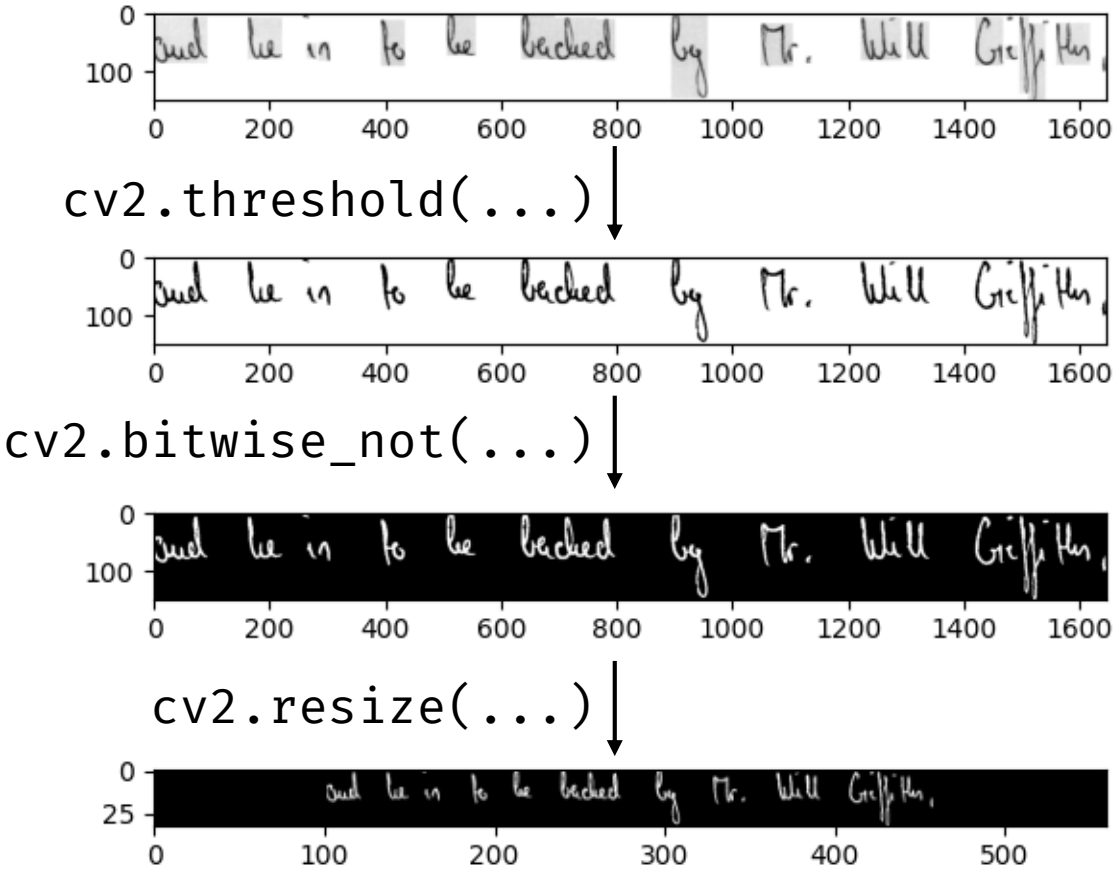
a01-000u-s00-03.png

sentences.txt

```
# format: a01-000u-s0-00 0 ok 154 19 408 746 1663 91
A|MOVE|to|stop|Mr.|Gaitskell|from
#
#     a01-000u-s0-00  -> sentence/line id for form a01-000u
#     0               -> sentence number within this form
#     ok              -> result of word segmentation
#                            ok: line is correctly segmented
#                            er: segmentation of line has one or more errors
#
#     154             -> graylevel to binarize line
#     19              -> number of components for this part of the sentence
#     408 746 1663 91 -> bounding box around for this part of the sentence
#                        in the x,y,w,h format
#
#     A|MOVE|to|stop|Mr.|Gaitskell|from
#                     -> transcription for this part of the sentence. word
#                        tokens are separated by the character |
#
a01-000u-s00-00 0 ok 154 19 408 746 1661 89 A|MOVE|to|stop|Mr.|Gaitskell|from
a01-000u-s00-01 0 ok 156 19 395 932 1850 105 nominating|any|more|Labour|life|Peers
a01-000u-s00-02 0 ok 157 16 408 1106 1986 105 is|to|be|made|at|a|meeting|of|Labour
a01-000u-s00-03 0 err 156 9 430 1290 733 66 M Ps|tomorrow|.
...
```

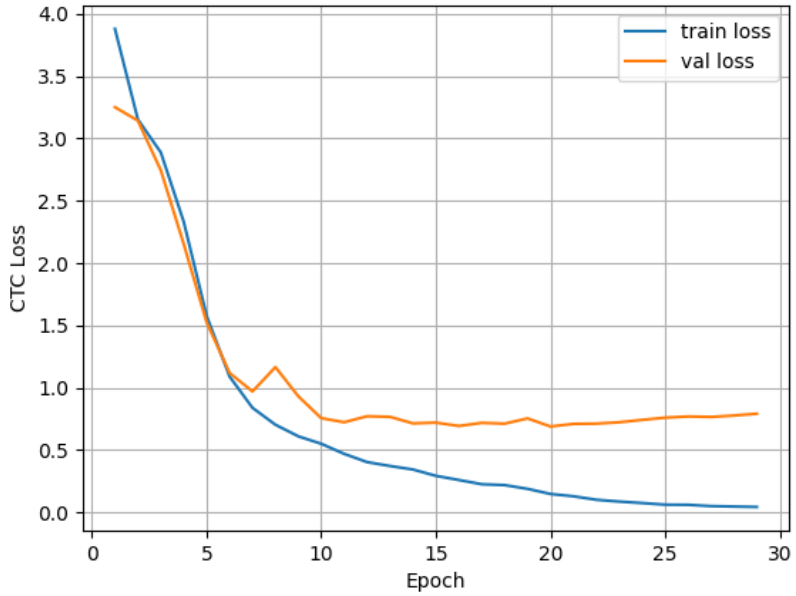
Препроцессинг датасета

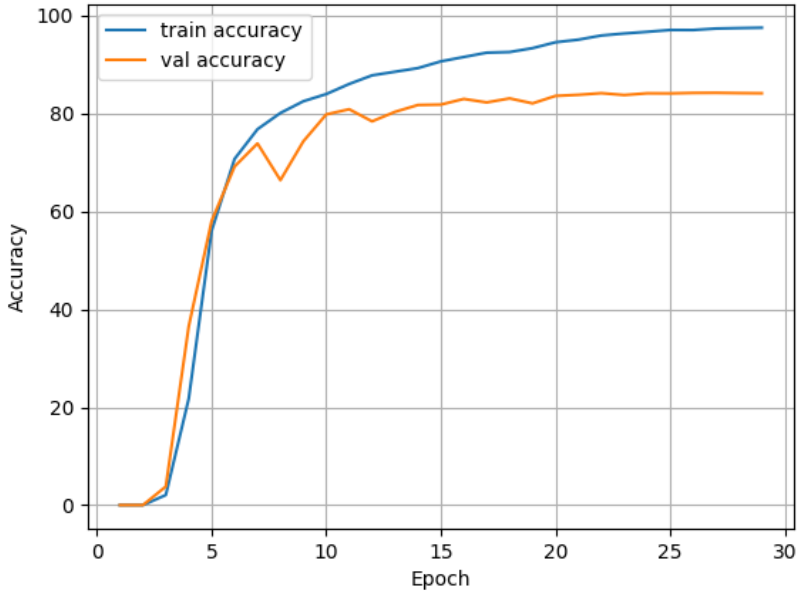
- Каждое изображение в датасете было преобразовано с помощью библиотеки OpenCV
  - Черно-белый канал 0-255
  - Размер 560x32
  - Чанки 4x32 пикселей, всего 140 чанков
- Полученные изображения сохраняются в бинарные файлы-дампы numpy-массивов
- Всего 14000 изображений
- Примерно 11500 тренировочных и 2500 валидационных изображений



Обучение

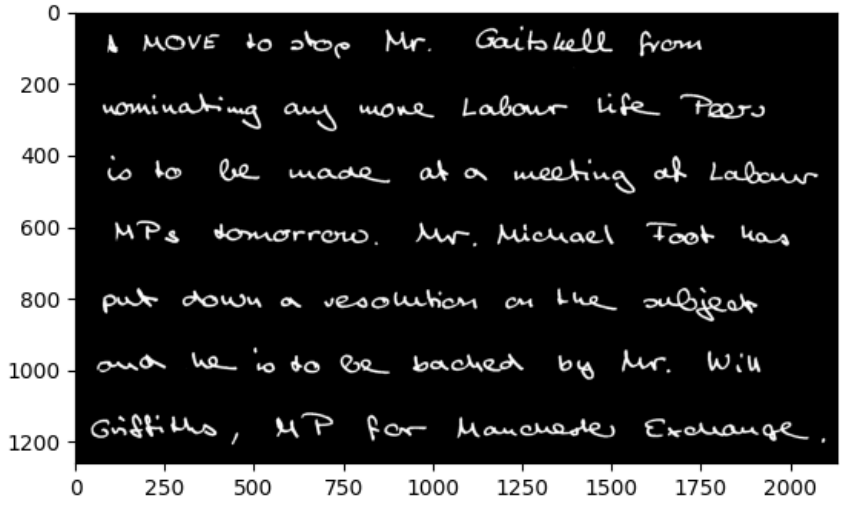
- Параметры обучения
  - Оптимизатор: AdamW
  - Скорость обучения (начальная): 0.001
  - Уменьшение скорости обучения при выходе val_loss на плато
  - Early stopping при отсутствии улучшения val_loss
  - Батчи по 64 изображения
  - Всего 30 эпох
- Результаты
  - Наилучшая по val_loss модель на 20-й эпохе
  - 94% точности на тренировочной выборке
  - 83% точность на валидационной выборке



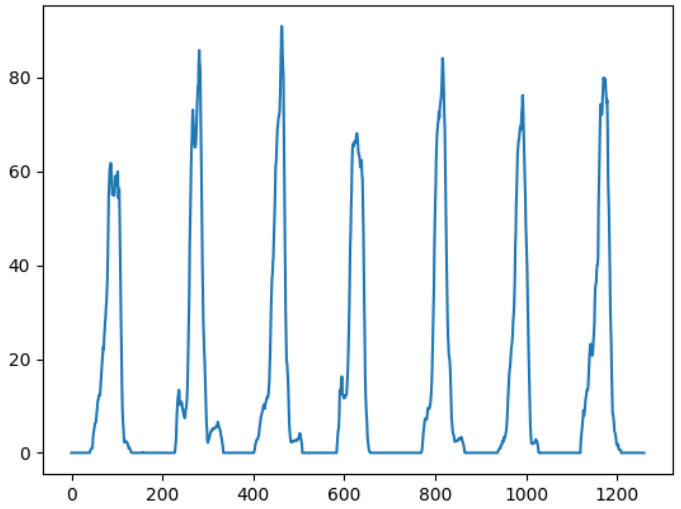


Разделение документа на строки

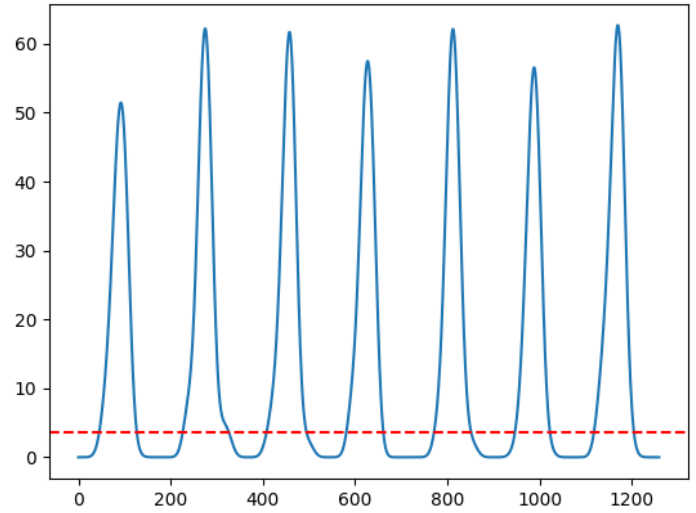
- Подсчитываются средние значения интенсивности пикселей по строкам изображения
- Выполняется сглаживание по Гауссу
- Классификация на светлые и темные участки с помощью `cv2.THRESH_OTSU`
- Каждой строке соответствует “светлый” + 2 соседних “темных” участка (межстрочное пространство, петельки от букв и т.п.)



Исходное изображение



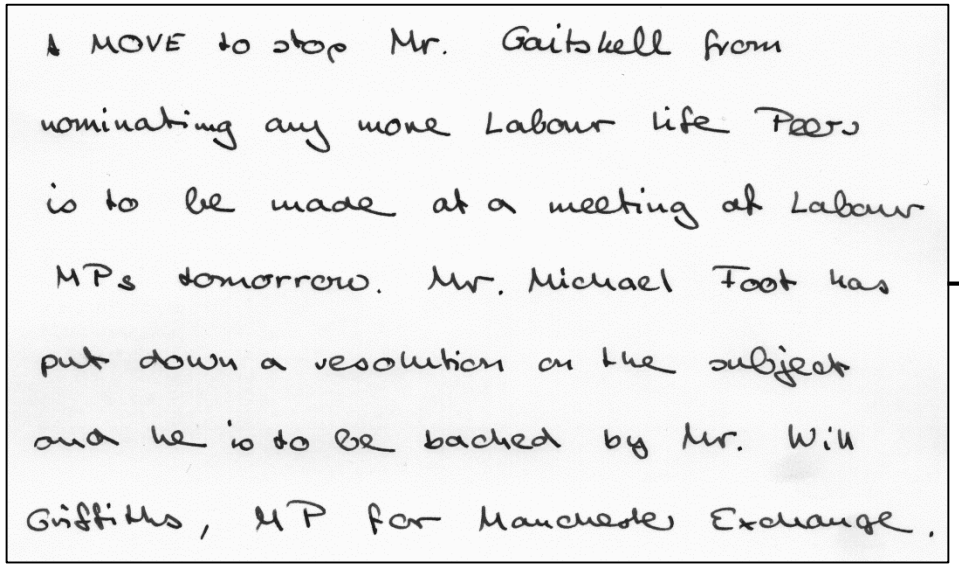
Построчная средняя интенсивность до сглаживания



Построчная средняя интенсивность после сглаживания (красный пунктир - порог)

Консольное python-приложение: пример из IAM

- Результатом реализации проекта стало CLI приложение на python, распознающее текст на картинке, переданной ему в качестве аргумента
- На примере ниже из датасета IAM с целыми формами (а не строками) можно видеть, что в целом модель неплохо распознает даже сложный почерк
- Там, где модель ошибается, ошибочные символы в исходном тексте зачастую действительно трудноразличимы и похожи на версию модели
- Таким образом, с визуальным распознаванием рукописного текста модель справляется неплохо




```
python app.py iam.png
```

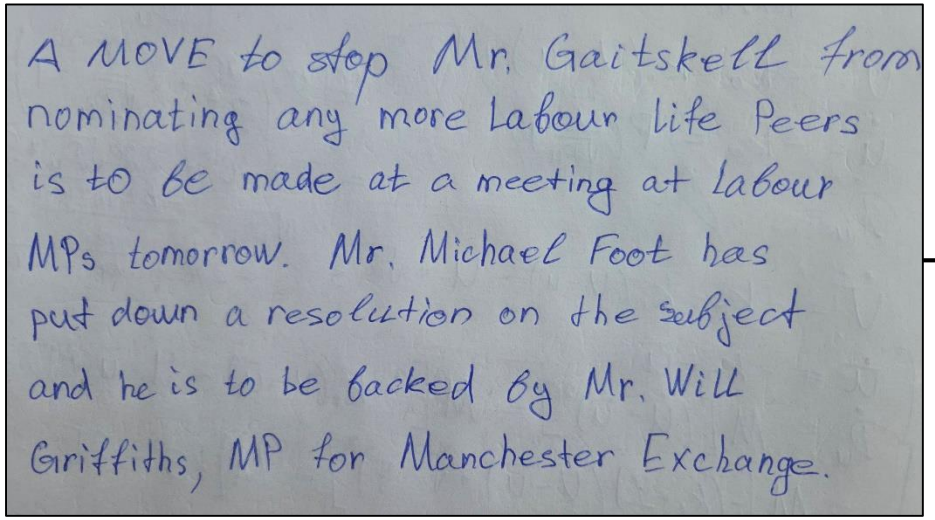
```
Loading model...
Model has been loaded successfully!

Recognition result:
t MOVE to stop Mr. Gaits kel from
nominatring any mone Labour life " Pizers
is to be made at a meting af Laborur
APg tomororw . Mhr. Michael " Fot has
put dow a vesotutions on the subject
ond he ' rs to be bached by Mr. TWil
Grifiths , M P for Mauchestes Exchauge .
```

CER = 15.75%, Accuracy = 84.25%

Консольное python-приложение: пример с собственным почерком

- Пример с собственным почерком показал, что модель неплохо справляется с относительно разборчивым почерком, который совсем “не видела” (в отличие от примера из IAM)
- Для узких букв или почерков с большими петлями разрешения в 32 пикселя по высоте не хватает, что прослеживается и на некоторых примерах из валидационных данных
- В промышленной реализации можно попробовать обучать на мощном девайсе задав разрешение 64 пикселя по высоте
- В остальном для распознавания рукописного текста точность 83.5% на таком примере является хорошим показателем и улучшается с помощью постобработки языковых моделями



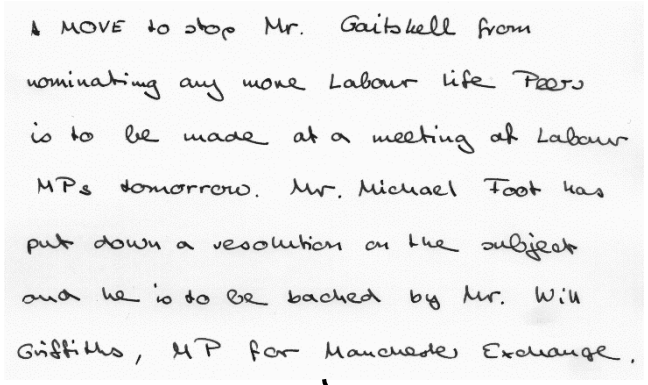

```
python app.py my.jpg
```

```
Loading model...
Model has been loaded successfully!

Recognition result:
A NovE to step Mr. Giaitskeltl frora
nominating any ' nore Laboun lite , Pers
is to be made at a neting at labour
wos tomorow. Mr. Michacl Fot has .
peit dowmn a resolution on the subject
and he is to be backed by Mr. wil
caritiths , me for Nanciester Erchange .
```

CER = 16.54%, Accuracy = 83.46%

Выводы



t MOVE to stop Mr. Gaits kel from nominatring any mone Labour life " Pizers is to be made at a meting af Laborur APg tomororw . Mhr. Michael " Fot has put dow a vesotutions on the subject ond he ' rs to be bached by Mr. TWil Grifiths , M P for Mauchestes Exchauge .

A MOVE to stop Mr. Gaitskel from nominating any more Labour life Peers is to be made at a meeting af Labour MPs tomorrow . Mr. Michael Foot has put down a resolutions on the subject and he is to be backed by Mr. Wil Griffiths , MP for Manchester Exchange .

Это слово похоже на “vesotutions”...

Но такого слова нет, видимо имелось в виду “resolutions”!

Разработанный проект

- Разработанные модель машинного обучения и приложение позволяют неплохо справляться с визуальным распознаванием рукописных текстов
- Расширение проекта до промышленной реализации должно включать в себя языковое решение по исправлению распознанных символов в контексте конкретного естественного языка